AMSTRAD Especial

REVISTA INDEPENDIENTE PARA USUARIOS DE ORDENADORES AMSTRAD
AÑO II N.º 4

350 ptas.
Canarias, Ceuta y Melilla 335 ptas.

BATMAN Y FAIRLIGHT DOS JUEGOS PARA TU PCW

PASCAL: UN LENGUAJE PARA APRENDER A HABLAR

PISTA A PISTA SOBRE EL DISCO

GESTION CONTABLE PARA EL PCW

ESPECIAL POKES: 10 JUEGOS EN LOS QUE VAS A GANAR

EN EL FONDO DEL CPM: TODOS SUS SECRETOS

ASI SE HACE UN PROGRAMA QUE JUEGA AL AJEDREZ

HOBBY PRESS

Director Editorial
José I. Gómez-Centurión
Director Ejecutivo
José M.ª Díaz
Redactor Jefe
Juan José Martínez
Diseño gráfico
Soledad Fungairiña
Colaboradores
Eduardo Ruiz
Javier Barceló
David Sopuerta
Robert Chatwin
Francisco Portalo
Pedro Sudón
Miguel Sepúlveda
Francisco Martín
Jesús Alonso
Pedro S. Pérez
Amalio Gómez
Secretaría Redacción
Carmen Santamaría
Fotografía
Carlos Candel
Portada
M. Barco
Ilustradores
J. Igual, J. Pons, F. L. Frontán, J. Septien, Pejo, J. J. Mora

Edita
HOBBY PRESS, S.A.

Presidente
María Andrino
Consejero Delegado
José I. Gómez-Centurión

Jefe de Producción
Carlos Peropadre

Marketing
Marta García
Jefe de Publicidad
Concha Gutiérrez
Publicidad Barcelona
José Galán Cortés
Tél: (93) 303 10 22/313 71 62

Secretaría de Dirección
Marisa Cogorro

Suscripciones
M.ª Rosa González
M.ª del Mar Calzada

Redacción, Administración y Publicidad
Ctra. de Irún km 12,400
(Fuencarral) 28049 Madrid
Teléfonos: Suscrip.: 734 65 00
Redacción: 734 70 12

Dto. Circulación
Paulino Blanco

Distribución
Coedis, S. A. Valencia, 245
Barcelona

Imprime
Gráficas Reunidos
Avda. Aragón, 56 (MADRID)
Fotocomposición
Novocomp, S.A.
Nicolás Morales, 38-40
Fotomecánica
GROF
Ezequiel Solano, 16
Depósito Legal:
M-5836-1986

Derechos exclusivos de la revista
COMPUTING with the AMSTRAD

Representante para Argentina, Chile, Uruguay y Paraguay, Cía. Americana de Ediciones, S.R.L. Sud América 1.532. Tel.: 21 24 64. 1209 BUENOS AIRES (Argentina).

Se solicitará control OJD

AMSTRAD
sumario

Año II • Número 4 • Enero 1987
Precio 350 ptas. Canarias, Ceuta y Melilla 335 ptas.

JUEGA AL AJEDREZ

Presentamos en esta ocasión el clásico juego del ajedrez para jugar entre dos personas, el ordenador lo único que hace es controlar los movimientos que realizan los jugadores.

Las instrucciones de manejo del programa son muy sencillas, ya que es totalmente autoexplicativo, no obstante, ahí van algunas breves notas sobre su manejo.

Después de mover una pieza no se puede volver atrás.

Si le hemos dicho la pieza que queremos mover, pero no el destino, podemos volver atrás dándole una jugada incorrecta.

Para hacer el enroque basta con darle la posición del rey y la casilla que va a ocupar. Para el rey blanco sería:

Desde E8 hasta G8: enroque corto.

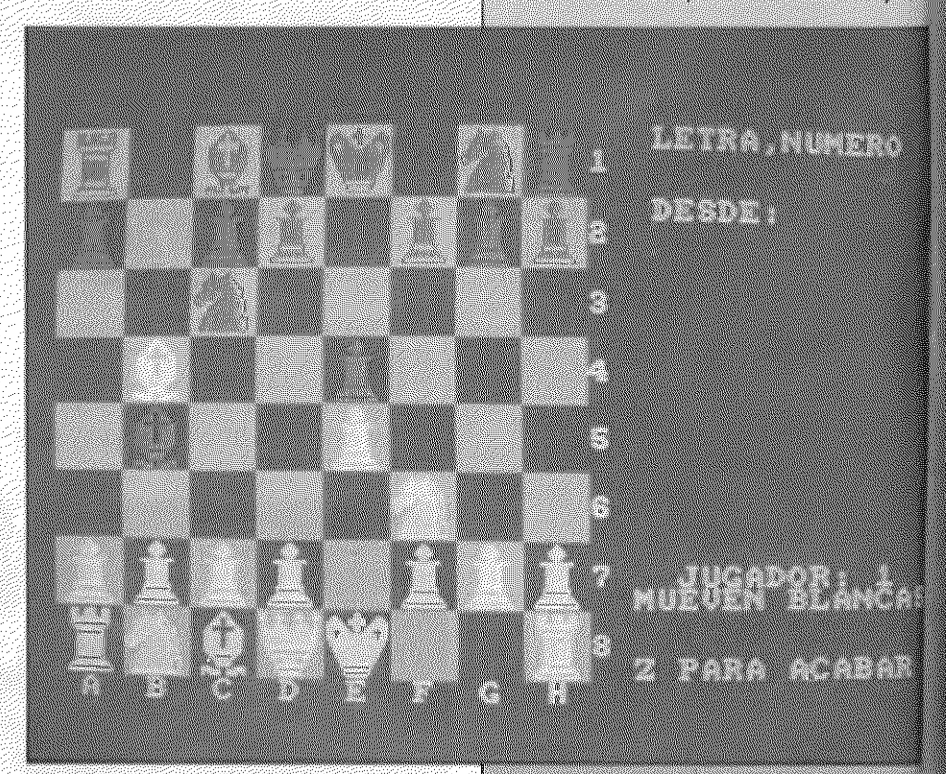
Desde E8 hasta C8: enroque largo.

En la línea 1800 están puestos los colores, puse esos porque en el monitor verde se ven mejor, para el de color poner los colores que queráis.

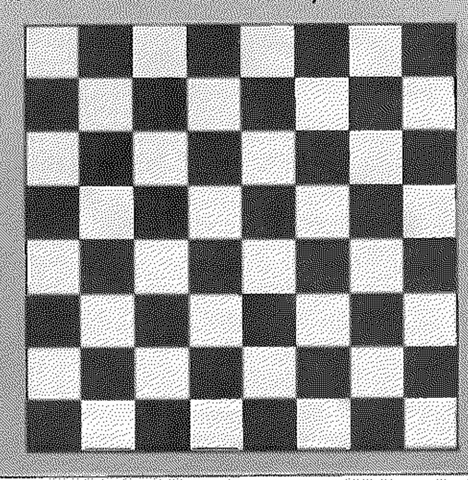
VARIABLES

JUGADA$ ()	Tabla de 500 elementos que contendrá las jugadas realizadas para poder ver la partida después de acabarla.
H$ ()	Tabla de 8×8, indica dónde se encuentran colocadas las piezas en el tablero.
A$	Coordenada horizontal de la pieza que se va a mover.
B$	Coordenada vertical de la pieza que se va a mover.
C$	Coordenada horizontal de la casilla donde se va a mover.
D$	Coordenada vertical de la casilla donde se va a mover.
TNI, TND, TBI, TBD, REYN, REYB	Cuando alguno de estos flag se pone a 1, indica que no se puede hacer el enroque con esa ficha.
Z$	Cuadro blanco.
X$	Cuadro negro.
T$, P$, AL$, CA$, R$, RN$	Contienen las figuras, torre, peón, alfil, caballo, rey y reina respectivamente.
PP	Puntero de la tabla JUGADA$().
REPETICION	Si está a 1, indica que se está viendo la partida repetida y no mete la jugada en la tabla JUGADA$().
JUGADOR	Jugador al que le toca mover.
FICHA$	Color de las piezas que tienen que mover.

```
110 CLEAR:DEFINT A-Z:GOSUB 1310:DIM J
UGADA$(500):JUGADOR=2:PEN 1
120 LOCATE 4,4:PRINT P$:LOCATE 10,4:P
RINT T$:LOCATE 16,4:PRINT CA$:LOCATE
22,4:PRINT AL$:LOCATE 28,4:PRINT R$:L
OCATE 34,4:PRINT RN$
125 LOCATE 12,25:PRINT CHR$(164)+" ER
 ZURI   1986"
130 LOCATE 6,2:PRINT"A J E D R E Z
 (2 JUGADORES)
140 MOVE 144,158:DRAW 144,240:DRAW 49
4,240:DRAW 494,158:DRAW 144,158:LOCAT
E 11,12:PRINT"1.- INSTRUCCIONES":LOCA
TE 11,14:PRINT"2.- EMPEZAR EL JUEGO"
150 A$=INKEY$:IF A$="1" THEN GOSUB 29
10 ELSE IF A$="2" THEN 160 ELSE 150
160 CLS:GOSUB 1890
170 H$(1,1)="TN":H$(2,1)="CN":H$(3,1)
="AN":H$(4,1)="RNN":H$(5,1)="RN":H$(6
,1)=H$(3,1):H$(7,1)=H$(2,1):H$(8,1)=H
$(1,1):FOR I=1 TO 8:H$(I,2)="PN":NEXT
180 FOR I=1 TO 8:H$(I,7)="PB":NEXT:H$
(1,8)="TB":H$(2,8)="CB":H$(3,8)="AB":
H$(4,8)="RNB":H$(5,8)="RB":H$(6,8)=H$
(3,8):H$(7,8)=H$(2,8):H$(8,8)=H$(1,8)
:PEN 1:LOCATE 28,1:PRINT"LETRA,NUMERO
":LOCATE 27,24:PRINT"Z PARA ACABAR"
190 PEN 1:IF JUGADOR=1 THEN JUGADOR=2
:FICHA$="NEGRAS" ELSE JUGADOR=1:FICHA
$="BLANCAS"
200 PRINT CHR$(22);CHR$(0):LOCATE 39,
21:PRINT"  ":LOCATE 34,4:PRINT"    ":L
OCATE 34,5:PRINT"    "
210 LOCATE 29,20:PRINT"JUGADOR:";JUGA
DOR:LOCATE 27,21:PRINT"MUEVEN ";FICHA
$:PRINT CHR$(22);CHR$(1):LOCATE 28,4:
```



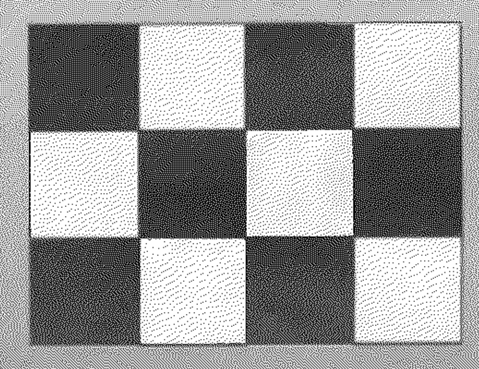
```
PRINT"DESDE:":A$=INKEY$:IF A$="" THEN 210
220 A$=UPPER$(A$):IF A$="Z" THEN 2220 ELSE IF A$<"A" OR A$>"H" THEN 210
230 IF A$="A" THEN A=1 ELSE IF A$="B" THEN A=2 ELSE IF A$="C" THEN A=3 ELSE IF A$="D" THEN A=4 ELSE IF A$="E" THEN A=5 ELSE IF A$="F" THEN A=6 ELSE IF A$="G" THEN A=7 ELSE IF A$="H" THEN A=8
240 LOCATE 34,4:PRINT A$;",":B$=INKEY$:IF B$="" THEN 240
250 IF B$<"1" OR B$>"8" THEN 240 ELSE LOCATE 36,4:PRINT B$
260 B=VAL(B$):IF RIGHT$(H$(A,B),1)="" THEN 270 ELSE 280
270 PRINT CHR$(7):LOCATE 29,14:PRINT"IMPOSIBLE":A$=INKEY$:IF A$="" THEN 270 ELSE PRINT CHR$(22);CHR$(0):LOCATE 29,14:PRINT STRING$(10," "):LOCATE 34,4:PRINT"   ":LOCATE 34,5:PRINT"   ":PRINT CHR$(22);CHR$(1):GOTO 210
280 IF (JUGADOR=1 AND RIGHT$(H$(A,B),1)="N") OR (JUGADOR=2 AND RIGHT$(H$(A,B),1)="B") THEN 270
290 LOCATE 28,5:PRINT"HASTA:":C$=INKEY$:IF C$="" THEN 290
300 C$=UPPER$(C$):IF C$<"A" OR C$>"H" THEN 290 ELSE LOCATE 34,5:PRINT C$;","
310 D$=INKEY$:IF D$<"1" OR D$>"8" THEN 310 ELSE LOCATE 36,5:PRINT D$
320 D=VAL(D$):IF C$="A" THEN C=1 ELSE IF C$="B" THEN C=2 ELSE IF C$="C" THEN C=3 ELSE IF C$="D" THEN C=4 ELSE IF C$="E" THEN C=5 ELSE IF C$="F" THEN C=6 ELSE IF C$="G" THEN C=7 ELSE IF C$="H" THEN C=8
330 IF H$(A,B)="PB" OR H$(A,B)="PN" THEN 350 ELSE IF H$(A,B)="TB" OR H$(A,B)="TN" THEN 590 ELSE IF H$(A,B)="AB" OR H$(A,B)="AN" THEN 770 ELSE IF H$(A,B)="CB" OR H$(A,B)="CN" THEN 1150
340 IF H$(A,B)="RB" OR H$(A,B)="RN" THEN 980 ELSE IF H$(A,B)="RNB" OR H$(A,B)="RNN" THEN 960
350 '***** PEON *****
360 IF A=C THEN 370 ELSE 430:'AVANZAR Y COMER
370 IF B>D AND H$(A,B)="PN" THEN 270:'NO PUEDEN IR PARA ATRAS LAS NEGRAS
380 IF D>B AND H$(A,B)="PB" THEN 270:'NO PUEDEN IR PARA ATRAS LAS BLANCAS
390 IF (B-2>D AND H$(A,B)="PB") OR (D-B>2 AND H$(A,B)="PN") THEN 270
400 IF (B-D=2 AND H$(A,B)="PB" AND B<>7) OR (D-B=2 AND H$(A,B)="PN" AND B<>2) THEN 270
410 IF (B+1=D OR B-1=D) AND H$(C,D)<>"" THEN 270
420 GOTO 500
430 IF A-1=C OR A+1=C THEN 440 ELSE 270
440 IF (A-1=C OR A+1=C) AND H$(C,D)="" THEN 270
450 IF (A-1=C OR A+1=C) AND H$(A,B)="PB" AND RIGHT$(H$(C,D),1)<>"N" THEN 270
460 IF (A-1=C OR A+1=C) AND H$(A,B)="PN" AND RIGHT$(H$(C,D),1)<>"B" THEN 270
470 IF B-1=D OR B+1=D THEN 480 ELSE 270
480 IF H$(A,B)="PB" AND RIGHT$(H$(C,D),1)<>"N" THEN 500
490 IF H$(A,B)="PN" AND RIGHT$(H$(C,D),1)<>"B" THEN 270
500 GOSUB 2400:IF SW=1 THEN SW=0:GOTO 270 ELSE GOSUB 1970
510 IF SW=2 THEN SW=0:BORDER 1,7 ELSE 530
520 A$=INKEY$:IF A$="" THEN 520 ELSE BORDER 1:PRINT CHR$(22);CHR$(0):LOCATE 31,14:PRINT"     "
530 IF D=1 THEN 540 ELSE IF D=8 THEN 550 ELSE 580:'CAMBIAMOS UN PEON POR UNA DAMA
540 IF C=1 OR C=3 OR C=5 OR C=7 THEN PEN 2:GOTO 560 ELSE PEN 0:GOTO 560
550 IF C=1 OR C=3 OR C=5 OR C=7 THEN PEN 0:GOTO 570 ELSE PEN 2:GOTO 570
560 LOCATE C+C+C-2,D+D+D-2:PRINT Z$:PEN 1:LOCATE C+C+C-2,D+D+D-2:PRINT RN$:H$(C,D)="RNB":GOTO 580
570 LOCATE C+C+C-2,D+D+D-2:PRINT Z$:LOCATE C+C+C-2,D+D+D-2:PEN 3:PRINT RN$:H$(C,D)="RNN"
580 GOTO 190
590 '***** TORRE *****
600 IF A=C THEN 610 ELSE IF B=D THEN 710 ELSE 270:'MOVIMIENTO VERTICAL Y HORIZONTAL
610 IF B<D THEN 620 ELSE 640
620 FOR I=B+1 TO D-1:IF H$(C,I)<>"" THEN 270 ELSE NEXT:IF (H$(A,B)="TB" OR H$(A,B)="RB" OR H$(A,B)="RNB") AND RIGHT$(H$(C,D),1)="B" THEN 270
630 FOR I=B+1 TO D-1:IF H$(C,I)<>"" THEN 270 ELSE NEXT:IF (H$(A,B)="TN" OR H$(A,B)="RN" OR H$(A,B)="RNN") AND RIGHT$(H$(C,D),1)="N" THEN 270 ELSE 660
640 FOR I=D+1 TO B-1:IF H$(C,I)<>"" THEN 270 ELSE NEXT:IF (H$(A,B)="TB" OR H$(A,B)="RB" OR H$(A,B)="RNB") AND RIGHT$(H$(C,D),1)="B" THEN 270
650 FOR I=D+1 TO B-1:IF H$(C,I)<>"" THEN 270 ELSE NEXT:IF (H$(A,B)="TN" OR
```



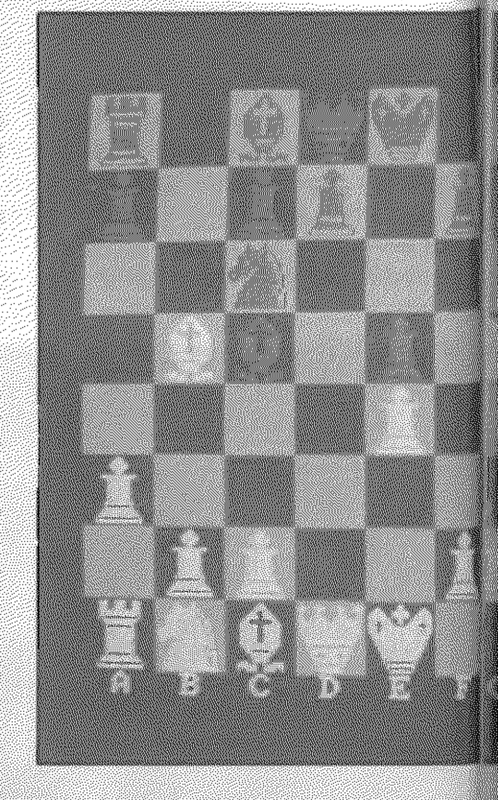
```
 H$(A,B)="RN" OR H$(A,B)="RNN") AND R
IGHT$(H$(C,D),1)="N" THEN 270
660 GOSUB 2400:IF SW=1 THEN SW=0:GOTO
 270 ELSE GOSUB 1970
670 IF SW=2 THEN SW=0:BORDER 1,7 ELSE
 690
680 A$=INKEY$:IF A$="" THEN 680 ELSE
BORDER 1:PRINT CHR$(22);CHR$(0):LOCAT
E 31,14:PRINT"     "
690 IF A=8 AND B=8 THEN TBD=1 ELSE IF
 A=1 AND B=8 THEN TBI=1 ELSE IF A=1 A
ND B=1 THEN TNI=1 ELSE TND=1
700 GOTO 190
710 IF A<C THEN 720 ELSE 740
720 FOR I=A+1 TO C-1:IF H$(I,D)<>"" T
HEN 270 ELSE NEXT:IF (H$(A,B)="TB" OR
 H$(A,B)="RB" OR H$(A,B)="RNB") AND R
IGHT$(H$(C,D),1)="B" THEN 270
```



```
730 FOR I=A+1 TO C-1:IF H$(I,D)<>"" T
HEN 270 ELSE NEXT:IF (H$(A,B)="TN" OR
 H$(A,B)="RN" OR H$(A,B)="RNN") AND R
IGHT$(H$(C,D),1)="N" THEN 270 ELSE 76
0
740 FOR I=C+1 TO A-1:IF H$(I,D)<>"" T
HEN 270 ELSE NEXT:IF (H$(A,B)="TN" OR
 H$(A,B)="RN" OR H$(A,B)="RNN") AND R
IGHT$(H$(C,D),1)="N" THEN 270
750 FOR I=C+1 TO A-1:IF H$(I,D)<>"" T
HEN 270 ELSE NEXT:IF (H$(A,B)="TB" OR
 H$(A,B)="RB" OR H$(A,B)="RNB") AND R
IGHT$(H$(C,D),1)="B" THEN 270
760 GOTO 660
770 '***** ALFIL *****
780 X=ABS(A-C):Y=ABS(B-D):IF X<>Y THE
N 270 ELSE X=A:Y=B
790 IF A<C AND B<D THEN 800 ELSE IF A
>C AND B<D THEN 830 ELSE IF A>C AND B
>D THEN 860 ELSE IF A<C AND B>D THEN
890 ELSE 270
800 X=X+1:Y=Y+1:IF X=C THEN 810 ELSE
IF H$(X,Y)<>"" THEN 270 ELSE 800
810 IF (H$(A,B)="AB" OR H$(A,B)="RB"
OR H$(A,B)="RNB") AND RIGHT$(H$(C,D),
1)="B" THEN 270
820 IF (H$(A,B)="AN" OR H$(A,B)="RN"
OR H$(A,B)="RNN") AND RIGHT$(H$(C,D),
1)="N" THEN 270 ELSE 920
830 X=X-1:Y=Y+1:IF X=C THEN 840 ELSE
IF H$(X,Y)<>"" THEN 270 ELSE 830
840 IF (H$(A,B)="AB" OR H$(A,B)="RB"
OR H$(A,B)="RNB") AND RIGHT$(H$(C,D),
1)="B" THEN 270
850 IF (H$(A,B)="AN" OR H$(A,B)="RN"
OR H$(A,B)="RNN") AND RIGHT$(H$(C,D),
1)="N" THEN 270 ELSE 920
860 X=X-1:Y=Y-1:IF X=C THEN 870 ELSE
IF H$(X,Y)<>"" THEN 270 ELSE 860
870 IF (H$(A,B)="AB" OR H$(A,B)="RB"
OR H$(A,B)="RNB") AND RIGHT$(H$(C,D),
1)="B" THEN 270
880 IF (H$(A,B)="AN" OR H$(A,B)="RN"
OR H$(A,B)="RNN") AND RIGHT$(H$(C,D),
1)="N" THEN 270 ELSE 920
890 X=X+1:Y=Y-1:IF X=C THEN 900 ELSE
IF H$(X,Y)<>"" THEN 270 ELSE 890
900 IF (H$(A,B)="AB" OR H$(A,B)="RB"
OR H$(A,B)="RNB") AND RIGHT$(H$(C,D),
1)="B" THEN 270
910 IF (H$(A,B)="AN" OR H$(A,B)="RN"
OR H$(A,B)="RNN") AND RIGHT$(H$(C,D),
1)="N" THEN 270
920 GOSUB 2400:IF SW=1 THEN SW=0:GOTO
 270 ELSE GOSUB 1970
930 IF SW=2 THEN SW=0:BORDER 1,7 ELSE
 950
940 A$=INKEY$:IF A$="" THEN 940 ELSE
BORDER 1:PRINT CHR$(22);CHR$(0):LOCAT
E 31,14:PRINT"     "
950 GOTO 190
960 '***** REINA *****
970 IF A=C OR B=D THEN 590 ELSE 770
980 '***** REY *****
990 IF (A=5 AND B=8 AND C=3 AND D=8)O
R(A=5 AND B=8 AND C=7 AND D=8)OR(A=5
AND B=1 AND C=3 AND D=1)OR(A=5 AND B=
1 AND C=7 AND D=1) THEN 1000 ELSE 112
0
1000 GOSUB 2400:IF SW=1 THEN SW=0:GOT
O 270
```



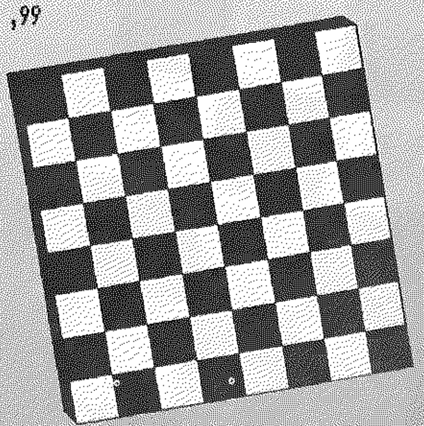
```
1010 IF SW=2 THEN SW=0:BORDER 1,7 ELS
E 1030
1020 A$=INKEY$:IF A$="" THEN 1020 ELS
E BORDER 1:PRINT CHR$(22);CHR$(0):LOC
ATE 31,14:PRINT"     "
1030 IF A=5 AND B=8 AND C=3 AND D=8 A
```



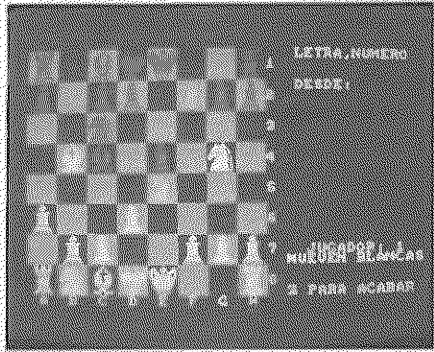

```
ND REYB=0 AND TBI=0 AND H$(2,8)="" AN
D H$(3,8)="" AND H$(4,8)="" THEN 1100
:'ENROQUE LARGO DE BLANCAS
1040 IF A=5 AND B=1 AND C=3 AND D=1 A
ND REYN=0 AND TNI=0 AND H$(2,1)="" AN
D H$(3,1)="" AND H$(4,1)="" THEN 1080
:'ENROQUE LARGO DE NEGRAS
1050 IF A=5 AND B=8 AND C=7 AND D=8 A
ND REYB=0 AND TBD=0 AND H$(7,8)="" AN
D H$(6,8)="" THEN 1090:'ENROQUE CORTO
 DE BLANCAS
1060 IF A=5 AND B=1 AND C=7 AND D=1 A
ND REYN=0 AND TND=0 AND H$(7,1)="" AN
D H$(6,1)="" THEN 1110:'ENROQUE CORTO
 DE NEGRAS
1070 GOTO 270
1080 TNI=1:REYN=1:PEN 2:LOCATE 13,1:P
RINT Z$:LOCATE 1,1:PRINT Z$:PEN 3:LOC
ATE 7,1:PRINT R$:LOCATE 10,1:PRINT T$
:H$(4,1)=H$(1,1):H$(3,1)=H$(5,1):H$(1
,1)="":H$(5,1)="":GOTO 190
1090 TBD=1:REYB=1:PEN 0:LOCATE 13,22:
PRINT Z$:PEN 1:LOCATE 19,22:PRINT R$:
PEN 2:LOCATE 22,22:PRINT Z$:PEN 1:LOC
ATE 16,22:PRINT T$:H$(7,8)=H$(5,8):H$
(6,8)=H$(8,8):H$(5,8)="":H$(8,8)="":G
OTO 190
1100 TBI=1:REYB=1:PEN 0:LOCATE 13,22:
PRINT Z$:LOCATE 1,22:PRINT Z$:PEN 1:L
OCATE 7,22:PRINT R$:LOCATE 10,22:PRIN
T T$:H$(4,8)=H$(1,8):H$(3,8)=H$(5,8):
H$(1,8)="":H$(5,8)="":GOTO 190
1110 TND=1:REYN=1:PEN 2:LOCATE 13,1:P
RINT Z$:PEN 3:LOCATE 19,1:PRINT R$:PE
N 0:LOCATE 22,1:PRINT Z$:PEN 3:LOCATE
```




```
 16,1:PRINT T$:H$(7,1)=H$(5,1):H$(6,1
)=H$(8,1):H$(5,1)="":H$(8,1)="":GOTO
190
1120 X=ABS(A-C):Y=ABS(B-D):LOCATE 27,
12:IF X>1 OR Y>1 THEN 270
1130 IF B=1 THEN REYN=1 ELSE REYB=1
1140 IF A=C OR B=D THEN 590 ELSE 770
1150 '***** CABALLO *****
1160 IF A-1=C AND B-2=D THEN 1250
1170 IF A-2=C AND B-1=D THEN 1250
1180 IF A-2=C AND B+1=D THEN 1250
1190 IF A-1=C AND B+2=D THEN 1250
1200 IF A+1=C AND B+2=D THEN 1250
1210 IF A+2=C AND B+1=D THEN 1250
1220 IF A+2=C AND B-1=D THEN 1250
1230 IF A+1=C AND B-2=D THEN 1250
1240 GOTO 270
1250 IF H$(A,B)="CB" AND RIGHT$(H$(C,
D),1)="B" THEN 270
1260 IF H$(A,B)="CN" AND RIGHT$(H$(C,
D),1)="N" THEN 270
1270 GOSUB 2400:IF SW=1 THEN SW=0:GOT
O 270 ELSE GOSUB 1970
1280 IF SW=2 THEN SW=0:BORDER 1,7 ELS
E 1300
1290 A$=INKEY$:IF A$="" THEN 1290 ELS
E BORDER 1:PRINT CHR$(22);CHR$(0):LOC
ATE 31,14:PRINT"     "
1300 GOTO 190
1310 SYMBOL AFTER 144
1320 SYMBOL 144,0,7,7,7,7,7,4,3
1330 SYMBOL 145,0,60,60,60,255,255,0,
255
1340 SYMBOL 146,0,224,224,224,224,224
,32,192
1350 SYMBOL 147,1,1,1,1,1,1,1,1
1360 SYMBOL 148,255,0,255,255,255,255
,255,255
1370 SYMBOL 149,128,128,128,128,128,1
28,128,128
1380 SYMBOL 150,1,1,1,3,7,8,15,0
1390 SYMBOL 151,255,255,0,255,255,0,2
55,0
1400 SYMBOL 152,128,128,128,192,224,1
6,240
1410 SYMBOL 153,0,0,0,0,0,0,0,3
1420 SYMBOL 154,0,24,60,126,126,60,24
,255
1430 SYMBOL 155,0,0,0,0,0,0,0,192
1440 SYMBOL 156,255,0,255,255,255,0,2
55,0
1450 SYMBOL 157,0,0,0,0,0,1,3,7
1460 SYMBOL 158,0,24,60,102,231,231,0
,0
1470 SYMBOL 159,0,0,0,0,0,128,192,224
1480 SYMBOL 160,7,7,7,7,7,7,7,3
1490 SYMBOL 161,231,231,231,231,231,2
31,255,255
1500 SYMBOL 162,224,224,224,224,224,2
24,224,192
1510 SYMBOL 163,1,0,0,0,15,15,9,0
1520 SYMBOL 192,129,255,126,126,36,23
1,231,0
1530 SYMBOL 165,128,0,0,0,240,240,144
,0
1540 SYMBOL 166,0,0,24,60,126,127,119
,99
```

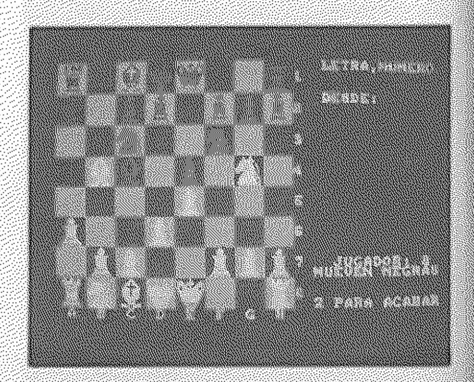


```
1550 SYMBOL 167,0,24,24,126,126,24,15
3,195
1560 SYMBOL 168,0,0,24,60,126,254,238
,198
1570 SYMBOL 169,119,119,63,63,31,31,1
5,15
1580 SYMBOL 170,238,238,252,252,248,2
48,240,240
1590 SYMBOL 171,7,7,3,2,1,1,0,0
1600 SYMBOL 172,255,255,255,0,255,0,2
55,0
1610 SYMBOL 173,224,224,192,64,128,12
8,0
1620 SYMBOL 174,0,1,1,3,7,15,15,14
1630 SYMBOL 175,0,4,140,254,253,252,2
```

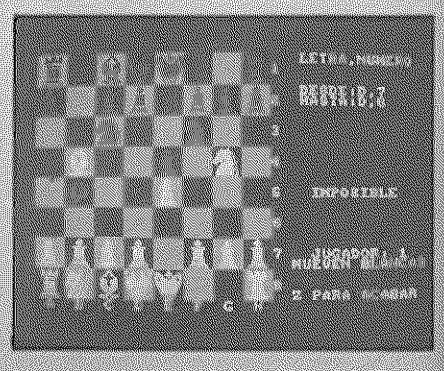



```
54,255
1640 SYMBOL 176,0,0,0,0,0,128,64,96
1650 SYMBOL 177,31,63,127,95,120,49,3
,3
1660 SYMBOL 178,255,223,191,127,255,2
55,255,255
1670 SYMBOL 179,32,160,160,144,208,20
8,208,144
1680 SYMBOL 180,7,15,15,15,31,31,31,0
1690 SYMBOL 181,255,255,255,255,255,2
55,255,0
1700 SYMBOL 182,160,184,200,200,200,2
00,248
1710 SYMBOL 183,0,32,113,32,33,35,55,
63
1720 SYMBOL 184,0,129,195,129,129,195
,231,255
1730 SYMBOL 185,0,4,142,4,132,196,236
,252
1740 SYMBOL 186,63,63,63,31,31,15,0,7
1750 SYMBOL 187,255,255,255,255,255,2
55,0,255
1760 SYMBOL 188,252,252,252,248,248,2
40,16,224
1770 SYMBOL 189,7,2,3,1,1,0,0,0
1780 SYMBOL 190,126,60,126,255,0,255,
255,0
1790 SYMBOL 191,224,64,192,128,128,0,
0,0
1800 MODE 1:INK 1,24:INK 2,14:INK 3,6
1810 z$=STRING$(3,CHR$(143))+CHR$(10)
+CHR$(8)+CHR$(8)+CHR$(8)+STRING$(3,CH
R$(143))+CHR$(10)+CHR$(8)+CHR$(8)+CHR
$(8)+STRING$(3,CHR$(143))
1820 x$=STRING$(3,CHR$(128))+CHR$(10)
+CHR$(8)+CHR$(8)+CHR$(8)+STRING$(3,CH
R$(128))+CHR$(10)+CHR$(8)+CHR$(8)+CHR
$(8)+STRING$(3,CHR$(128))
1830 T$=CHR$(144)+CHR$(145)+CHR$(146)
+CHR$(10)+CHR$(8)+CHR$(8)+CHR$(8)+CHR
$(147)+CHR$(148)+CHR$(149)+CHR$(10)+C
HR$(8)+CHR$(8)+CHR$(8)+CHR$(150)+CHR$
(151)+CHR$(152)
1840 P$=CHR$(153)+CHR$(154)+CHR$(155)
+CHR$(10)+CHR$(8)+CHR$(8)+CHR$(143)+C
HR$(10)+CHR$(8)+CHR$(8)+CHR$(150)+CHR
$(156)+CHR$(152)
1850 AL$=CHR$(157)+CHR$(158)+CHR$(159
)+CHR$(10)+CHR$(8)+CHR$(8)+CHR$(8)+CH
```

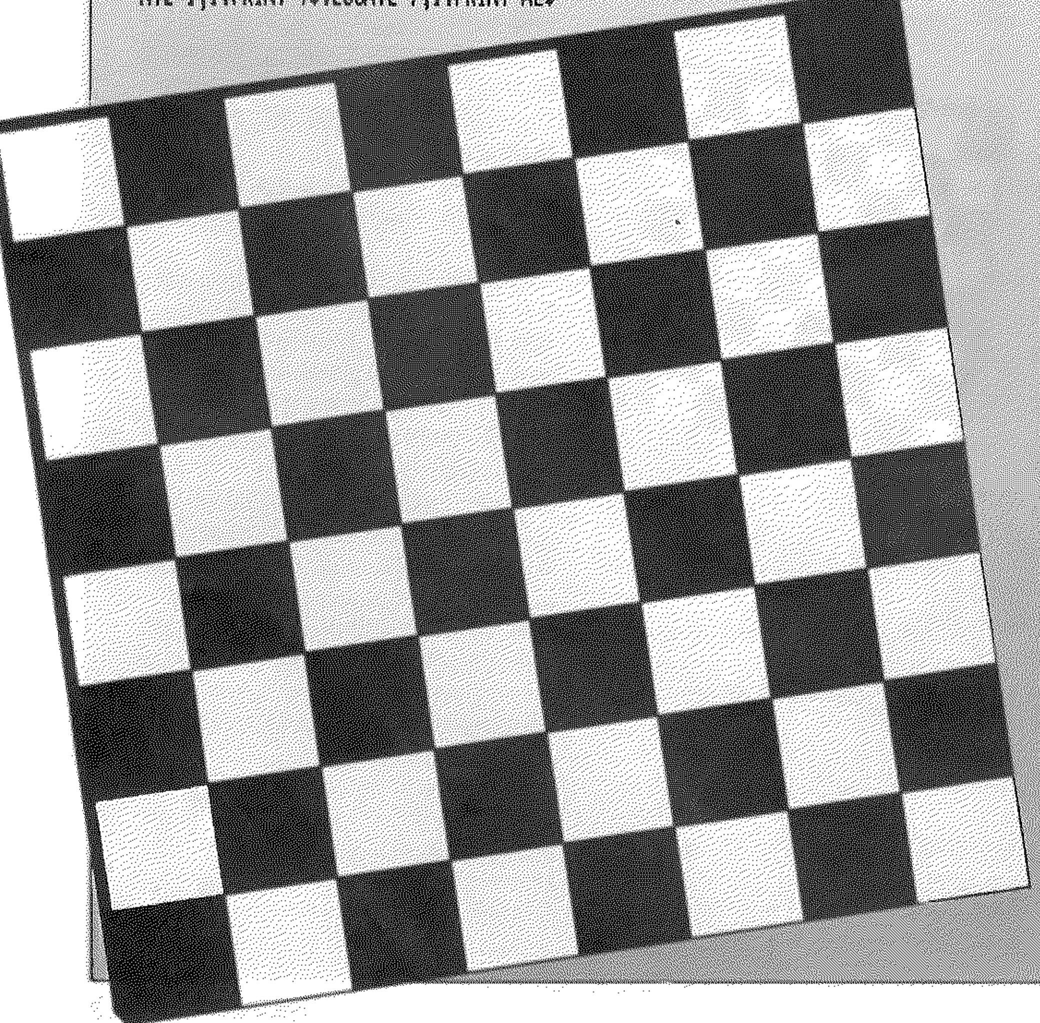
```
R$(160)+CHR$(161)+CHR$(162)+CHR$(10)+
CHR$(8)+CHR$(8)+CHR$(8)+CHR$(163)+CHR
$(192)+CHR$(165)
1860 R$=CHR$(166)+CHR$(167)+CHR$(168)
+CHR$(10)+CHR$(8)+CHR$(8)+CHR$(8)+CHR
$(169)+CHR$(143)+CHR$(170)+CHR$(10)+C
HR$(8)+CHR$(8)+CHR$(8)+CHR$(171)+CHR$
(172)+CHR$(173)
1870 CA$=CHR$(174)+CHR$(175)+CHR$(176
)+CHR$(10)+CHR$(8)+CHR$(8)+CHR$(8)+CH
R$(177)+CHR$(178)+CHR$(179)+CHR$(10)+
CHR$(8)+CHR$(8)+CHR$(8)+CHR$(180)+CHR
$(181)+CHR$(182)
1880 RN$=CHR$(183)+CHR$(184)+CHR$(185
)+CHR$(10)+CHR$(8)+CHR$(8)+CHR$(8)+CH
R$(186)+CHR$(187)+CHR$(188)+CHR$(10)+
CHR$(8)+CHR$(8)+CHR$(8)+CHR$(189)+CHR
$(190)+CHR$(191):RETURN
1890 PEN 2:FOR j=1 TO 24 STEP 6:FOR i
=1 TO 23 STEP 6:LOCATE i,j:PRINT z$:L
OCATE i+3,j:PRINT x$:NEXT:NEXT
1900 FOR j=4 TO 24 STEP 6:FOR i=1 TO
24 STEP 6:LOCATE i,j:PRINT x$:LOCATE
i+3,j:PRINT z$:NEXT:NEXT:K$=" 1  2  3
  4  5  6  7  8":PEN 1:LOCATE 1,25:PR
INT " A  B  C  D  E  F  G  H":FOR I=2
 TO 24 STEP 3:LOCATE 25,I:PRINT MID$(
K$,I,1):NEXT
1910 PEN 3:PRINT CHR$(22);CHR$(1):LOC
ATE 1,1:PRINT T$:LOCATE 7,1:PRINT AL$
:LOCATE 10,1:PRINT RN$:LOCATE 4,1:PRI
NT CA$:LOCATE 13,1:PRINT R$
1920 LOCATE 16,1:PRINT AL$:LOCATE 19,
1:PRINT CA$:LOCATE 22,1:PRINT T$:FOR
I=1 TO 22 STEP 3:LOCATE I,4:PRINT P$:
NEXT
1930 PEN 1:LOCATE 1,22:PRINT T$:LOCAT
E 7,22:PRINT AL$:LOCATE 10,22:PRINT R
N$:LOCATE 4,22:PRINT CA$:LOCATE 13,22
:PRINT R$
1940 LOCATE 16,22:PRINT AL$:LOCATE 19
,22:PRINT CA$:LOCATE 22,22:PRINT T$:F
OR I=1 TO 22 STEP 3:LOCATE I,19:PRINT
 P$:NEXT
1950 RETURN
1960 'BORRADO E IMPRESION DE FIGURAS
1970 IF (A=1 OR A=3 OR A=5 OR A=7) AN
D (B=1 OR B=3 OR B=5 OR B=7) THEN 200
0
1980 IF (A=2 OR A=4 OR A=6 OR A=8) AN
D (B=2 OR B=4 OR B=6 OR B=8) THEN 200
0
1990 PEN 0:LOCATE A+A+A-2,B+B+B-2:PRI
NT Z$:GOTO 2010:'CUADRO NEGRO
2000 PEN 2:LOCATE A+A+A-2,B+B+B-2:PRI
NT Z$:'CUADRO BLANCO
2010 IF (C=1 OR C=3 OR C=5 OR C=7) AN
D (D=1 OR D=3 OR D=5 OR D=7) THEN 204
0
2020 IF (C=2 OR C=4 OR C=6 OR C=8) AN
D (D=2 OR D=4 OR D=6 OR D=8) THEN 204
0
2030 PEN 0:LOCATE C+C+C-2,D+D+D-2:PRI
NT Z$:GOTO 2050:'CUADRO NEGRO
2040 PEN 2:LOCATE C+C+C-2,D+D+D-2:PRI
NT Z$:'CUADRO BLANCO
2050 PEN 1:LOCATE C+C+C-2,D+D+D-2
2060 IF H$(A,B)="PB" THEN PRINT P$:GO
TO 2180
```



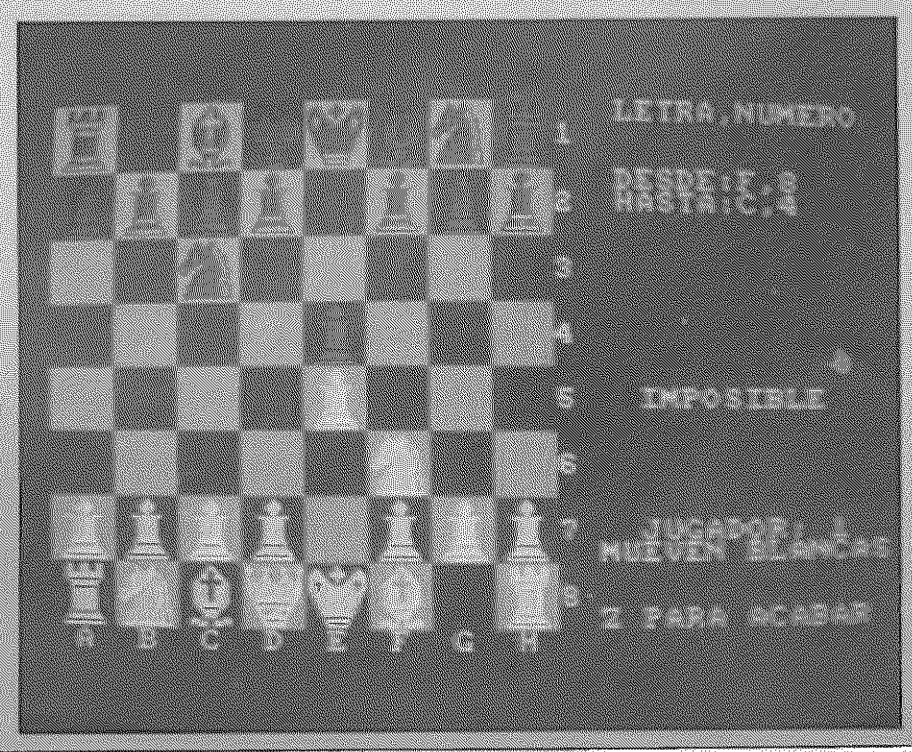

```
2070 IF H$(A,B)="PN" THEN PEN 3:PRINT
 P$:GOTO 2180
2080 IF H$(A,B)="TB" THEN PRINT T$:GO
TO 2180
2090 IF H$(A,B)="TN" THEN PEN 3:PRINT
 T$:GOTO 2180
2100 IF H$(A,B)="CB" THEN PRINT CA$:G
OTO 2180
2110 IF H$(A,B)="CN" THEN PEN 3:PRINT
 CA$:GOTO 2180
2120 IF H$(A,B)="AB" THEN PRINT AL$:G
OTO 2180
```

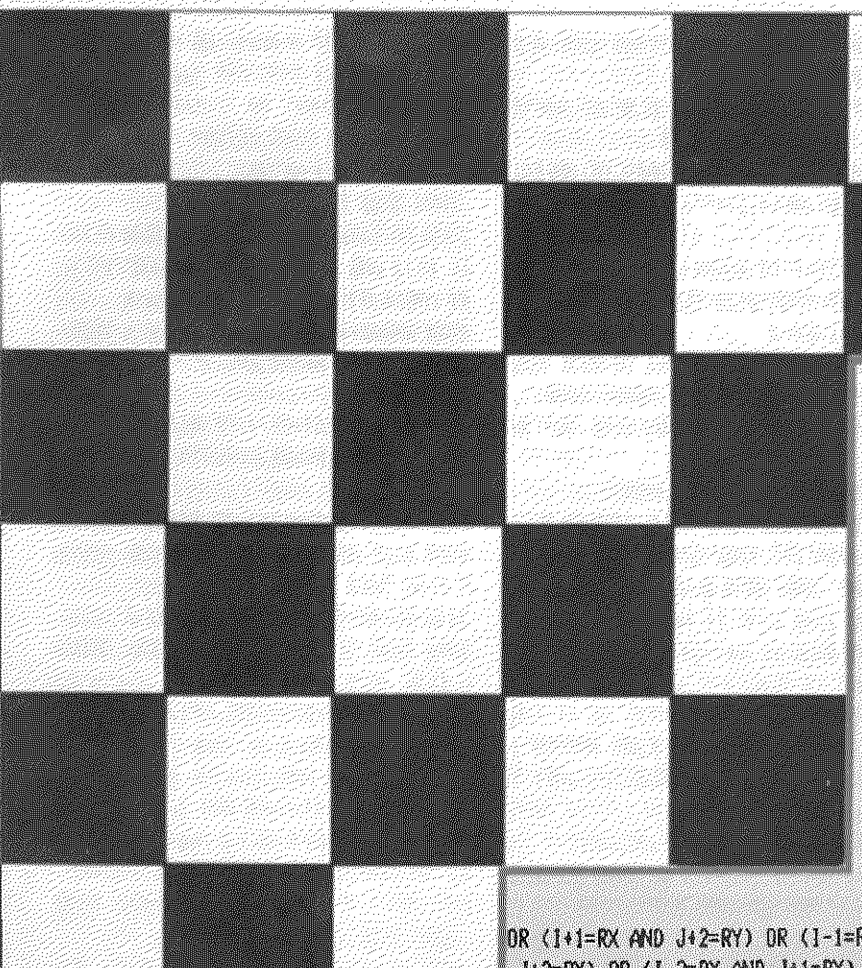



```
2130 IF H$(A,B)="AN" THEN PEN 3:PRINT
 AL$:GOTO 2180
2140 IF H$(A,B)="RB" THEN PRINT R$:GO
TO 2180
2150 IF H$(A,B)="RN" THEN PEN 3:PRINT
 R$:GOTO 2180
2160 IF H$(A,B)="RNB" THEN PRINT RN$:
GOTO 2180
2170 IF H$(A,B)="RNN" THEN PEN 3:PRIN
T RN$
2180 PRINT CHR$(22);CHR$(0):LOCATE 28
,4:PRINT STRING$(10," "):LOCATE 28,5:
PRINT STRING$(10," "):PRINT CHR$(22);
```

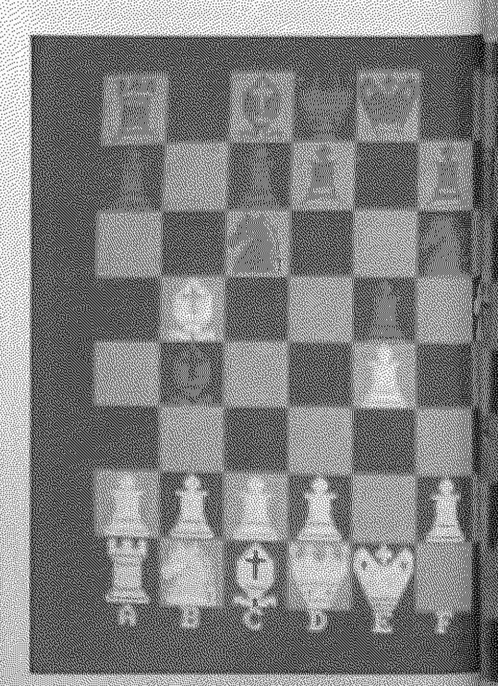


```
CHR$(1):PEN 1:IF REPETICION=1 THEN 22
10
2190 PP=PP+1:A$=STR$(A):B$=STR$(B):C$
=STR$(C):D$=STR$(D):JUGADA$(PP)=A$+B$
+C$+D$
2200 IF H$(C,D)="RN" OR H$(C,D)="RB"
THEN 2220
2210 H$(C,D)=H$(A,B):H$(A,B)="":RETUR
N
2220 REPETICION=1:WINDOW#1,26,40,1,24
:CLS#1:LOCATE#1,3,5:PRINT#1,"QUEREIS
VER":LOCATE#1,3,7:PRINT#1,"LA PARTIDA
?":LOCATE#1,6,9:PRINT#1,"S/N"
2230 A$=INKEY$:IF A$="" THEN 2230
2240 A$=UPPER$(A$):IF A$="S" THEN 225
0 ELSE IF A$="N" THEN 2370 ELSE 2230
2250 H$(1,1)="TN":H$(2,1)="CN":H$(3,1
)="AN":H$(4,1)="RNN":H$(5,1)="RN":H$(
6,1)=H$(3,1):H$(7,1)=H$(2,1):H$(8,1)=
H$(1,1):FOR I=1 TO 8:H$(I,2)="PN":NEX
T
2260 FOR I=1 TO 8:H$(I,7)="PB":NEXT:H
$(1,8)="TB":H$(2,8)="CB":H$(3,8)="AB"
:H$(4,8)="RNB":H$(5,8)="RB":H$(6,8)=H
$(3,8):H$(7,8)=H$(2,8):H$(8,8)=H$(1,8
)
2270 CLS:GOSUB 1890:JUGADOR=2
2280 FOR I=1 TO PP:IF JUGADOR=2 THEN
JUGADOR=1:FICHA$="BLANCAS" ELSE JUGAD
OR=2:FICHA$="NEGRAS"
2290 PRINT CHR$(22);CHR$(0):LOCATE 29
,2:PRINT"JUGADOR";JUGADOR:LOCATE 27,3
:PRINT"MUEVEN ";FICHA$:LOCATE 28,9:PR
INT"JUGADA";I
2300 A$=LEFT$(JUGADA$(I),2):A=VAL(A$)
:IF A=1 THEN A$="A" ELSE IF A=2 THEN
A$="B" ELSE IF A=3 THEN A$="C" ELSE I
F A=4 THEN A$="D" ELSE IF A=5 THEN A$
="E" ELSE IF A=6 THEN A$="F" ELSE IF
A=7 THEN A$="G" ELSE IF A=8 THEN A$="
H"
2310 B$=MID$(JUGADA$(I),3,2):B=VAL(B$
):LOCATE 28,11:PRINT"DESDE ";A$;B$
2320 C$=MID$(JUGADA$(I),5,2):C=VAL(C$
):IF C=1 THEN C$="A" ELSE IF C=2 THEN
 C$="B" ELSE IF C=3 THEN C$="C" ELSE
IF C=4 THEN C$="D" ELSE IF C=5 THEN C
$="E" ELSE IF C=6 THEN C$="F" ELSE IF
 C=7 THEN C$="G" ELSE IF C=8 THEN C$=
"H"
2330 D$=RIGHT$(JUGADA$(I),2):D=VAL(D$
):LOCATE 28,12:PRINT"HASTA ";C$;D$:PR
INT CHR$(22);CHR$(1)
2340 GOSUB 1970
2350 L$=INKEY$:IF L$="" THEN 2350 ELS
E NEXT
2360 GOTO 2220
2370 CLS#1:LOCATE 6,12:PRINT"QUEREIS
JUGAR":LOCATE 6,13:PRINT"OTRA PARTIDA
?"
2380 A$=INKEY$:IF A$="" THEN 2380 ELS
E A$=UPPER$(A$)
2390 IF A$="S" THEN 110 ELSE IF A$="N
" THEN CLS:NEW ELSE 2390
2400 ANT$=H$(A,B):POST$=H$(C,D):H$(C,
D)=H$(A,B):H$(A,B)="":'JAQUE AL REY B
LANCO
2410 FOR I=1 TO 8:FOR J=1 TO 8:IF H$(
I,J)="RB" THEN 2420 ELSE NEXT:NEXT
2420 RX=I:RY=J
2430 FOR I=1 TO 8:FOR J=1 TO 8:IF RIG
HT$(H$(I,J),1)="N" THEN GOSUB 2460
2440 IF SW<>0 THEN RETURN ELSE NEXT:N
EXT
2450 GOTO 2660:'NO HAY JAQUE
2460 IF H$(I,J)="PN" THEN 2470 ELSE I
F H$(I,J)="TN" THEN 2480 ELSE IF H$(I
,J)="CN" THEN 2550 ELSE IF H$(I,J)="A
N" THEN 2560 ELSE IF H$(I,J)="RNN" TH
EN 2620 ELSE IF H$(I,J)="RN" THEN 263
0
2470 IF (I+1=RX AND J+1=RY) OR (I-1=R
X AND J+1=RY) THEN 2640 ELSE RETURN
2480 IF I=RX THEN 2490 ELSE IF J=RY T
HEN 2520 ELSE RETURN
2490 IF J<RY THEN 2500 ELSE 2510
2500 FOR L=J+1 TO RY-1:IF H$(I,L)<>""
 THEN RETURN ELSE NEXT:GOTO 2640
2510 FOR L=RY+1 TO J-1:IF H$(I,L)<>""
 THEN RETURN ELSE NEXT:GOTO 2640
2520 IF I<RX THEN 2530 ELSE 2540
2530 FOR L=I+1 TO RX-1:IF H$(L,J)<>""
 THEN RETURN ELSE NEXT:GOTO 2640
2540 FOR L=RX+1 TO I-1:IF H$(L,J)<>""
 THEN RETURN ELSE NEXT:GOTO 2640
2550 IF (I+1=RX AND J-2=RY) OR (I+2=R
X AND J-1=RY) OR (I+2=RX AND J+1=RY)
OR (I+1=RX AND J+2=RY) OR (I-1=RX AND
 J+2=RY) OR (I-2=RX AND J+1=RY) OR (I
-2=RX AND J-1=RY) OR (I-1=RX AND J-2=
RY) THEN 2640 ELSE RETURN
2560 AX=I:AY=J:IF ABS(I-RX)=ABS(J-RY)
 THEN 2570 ELSE RETURN
2570 IF I<RX AND J>RY THEN 2580 ELSE
IF RX<I AND RY>J THEN 2590 ELSE IF I<
RX AND J<RY THEN 2600 ELSE 2610
2580 AX=AX+1:AY=AY-1:IF AX>RX-1 THEN
2640 ELSE IF H$(AX,AY)<>"" THEN RETUR
N ELSE 2580
2590 AX=AX-1:AY=AY+1:IF AX<RX+1 THEN
2640 ELSE IF H$(AX,AY)<>"" THEN RETUR
N ELSE 2590
2600 AX=AX+1:AY=AY+1:IF AX>RX-1 THEN
2640 ELSE IF H$(AX,AY)<>"" THEN RETUR
N ELSE 2600
2610 AX=AX-1:AY=AY-1:IF AX<RX+1 THEN
2640 ELSE IF H$(AX,AY)<>"" THEN RETUR
N ELSE 2610
2620 IF ABS(I-RX)=ABS(J-RY) THEN AX=I
:AY=J:GOTO 2570 ELSE 2480
2630 IF (I=RX AND J-1=RY) OR (I=RX AN
D J+1=RY) OR (I-1=RX AND J-1=RY) OR (
I-1=RX AND J=RY) OR (I-1=RX AND J+1=R
Y) OR (I+1=RX AND J-1=RY) OR (I+1=RX
AND J=RY) OR (I-1=RX AND J+1=RY) THEN
 2640 ELSE RETURN
2640 IF JUGADOR=1 THEN SW=1 ELSE LOCA
TE 31,14:PRINT"JAQUE":SW=2
2650 H$(A,B)=ANT$:H$(C,D)=POST$:RETUR
N
2660 FOR I=1 TO 8:FOR J=1 TO 8:IF H$(
I,J)="RN" THEN 2670 ELSE NEXT:NEXT
2670 RX=I:RY=J
2680 FOR I=1 TO 8:FOR J=1 TO 8:IF RIG
```



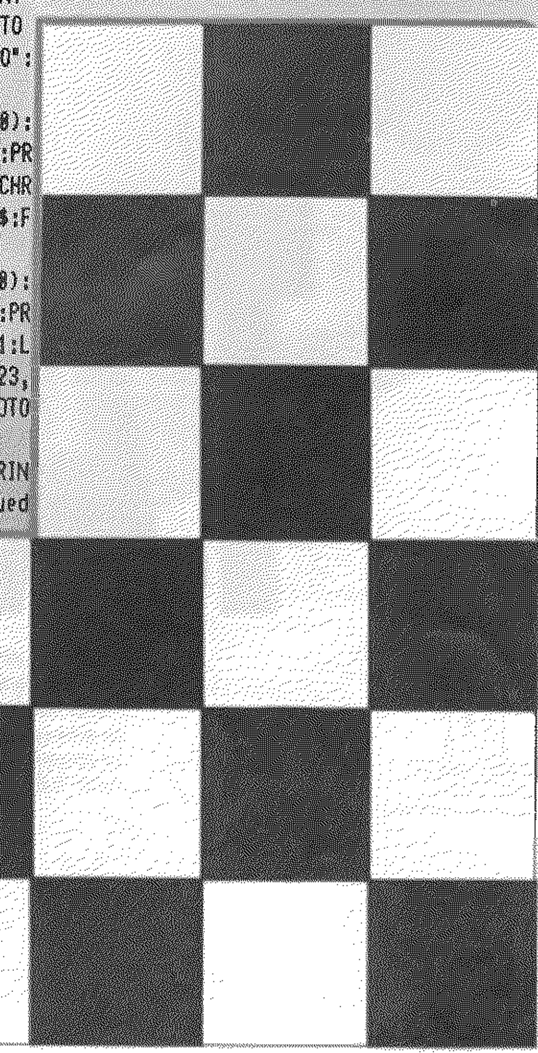


```
HT$(H$(I,J),1)="B" THEN GOSUB 2710
2690 IF SW<>0 THEN RETURN ELSE NEXT:N
EXT
2700 H$(A,B)=ANT$:H$(C,D)=POST$:RETUR
N:'NO HAY JAQUE
2710 IF H$(I,J)="PB" THEN 2720 ELSE I
F H$(I,J)="TB" THEN 2730 ELSE IF H$(I
,J)="CB" THEN 2800 ELSE IF H$(I,J)="A
B" THEN 2810 ELSE IF H$(I,J)="RNB" TH
EN 2870 ELSE IF H$(I,J)="RB" THEN 288
0
2720 IF (I+1=RX AND J+1=RY) OR (I-1=R
X AND J+1=RY) THEN 2890 ELSE RETURN
2730 IF I=RX THEN 2740 ELSE IF J=RY T
HEN 2770 ELSE RETURN
2740 IF J<RY THEN 2750 ELSE 2760
2750 FOR L=J+1 TO RY-1:IF H$(I,L)<>""
 THEN RETURN ELSE NEXT:GOTO 2890
2760 FOR L=RY+1 TO J-1:IF H$(I,L)<>""
 THEN RETURN ELSE NEXT:GOTO 2890
2770 IF I<RX THEN 2780 ELSE 2790
2780 FOR L=I+1 TO RX-1:IF H$(L,J)<>""
 THEN RETURN ELSE NEXT:GOTO 2890
2790 FOR L=RX+1 TO I-1:IF H$(L,J)<>""
 THEN RETURN ELSE NEXT:GOTO 2890
2800 IF (I+1=RX AND J-2=RY) OR (I+2=R
X AND J-1=RY) OR (I+2=RX AND J+1=RY)
OR (I+1=RX AND J+2=RY) OR (I-1=RX AND
 J+2=RY) OR (I-2=RX AND J+1=RY) OR (I
-2=RX AND J-1=RY) OR (I-1=RX AND J-2=
RY) THEN 2890 ELSE RETURN
2810 AX=I:AY=J:IF ABS(I-RX)=ABS(J-RY)
 THEN 2820 ELSE RETURN
2820 IF I<RX AND J>RY THEN 2830 ELSE
IF RX<I AND RY>J THEN 2840 ELSE IF I<
RX AND J<RY THEN 2850 ELSE 2860
2830 AX=AX+1:AY=AY-1:IF AX>RX-1 THEN
2890 ELSE IF H$(AX,AY)<>"" THEN RETUR
N ELSE 2830
2840 AX=AX-1:AY=AY+1:IF AX<RX+1 THEN
2890 ELSE IF H$(AX,AY)<>"" THEN RETUR
N ELSE 2840
2850 AX=AX+1:AY=AY+1:IF AX>RX-1 THEN
2890 ELSE IF H$(AX,AY)<>"" THEN RETUR
N ELSE 2850
2860 AX=AX-1:AY=AY-1:IF AX<RX+1 THEN
2890 ELSE IF H$(AX,AY)<>"" THEN RETUR
N ELSE 2860
2870 IF ABS(I-RX)=ABS(J-RY) THEN AX=I
:AY=J:GOTO 2820 ELSE 2730
2880 IF (I=RX AND J-1=RY) OR (I=RX AN
D J+1=RY) OR (I-1=RX AND J-1=RY) OR (
I-1=RX AND J=RY) OR (I-1=RX AND J+1=R
Y) OR (I+1=RX AND J-1=RY) OR (I+1=RX
AND J=RY) OR (I-1=RX AND J+1=RY) THEN
 2890 ELSE RETURN
2890 H$(A,B)=ANT$:H$(C,D)=POST$:IF JU
GADOR=2 THEN SW=1 ELSE LOCATE 31,14:P
RINT"JAQUE":SW=2
2900 RETURN
2910 CLS:LOCATE 19,1:PRINT P$:PRINT:P
RINT" El peon se muve verticalmente a
 lo lar-go de la columna en que se ha
lla situado  Al contrario de lo que o
curre con las restantes piezas,el peo
```




```
n siempre avanza,nunca puede retroced
er.
2920 PRINT"Si es la primera vez que s
e mueve el pe-on, este puede avanzar
dos casillas, en los demas casos solo
 una.
2930 PRINT"El peon captura en diagona
l avazando si-empre un paso, esto da
la oportunidad al peon de cambiar de
columna. Si el peon encuentra alguna
pieza frente a si,  se dice que esta
bloqueada ya que no puede avanzar
2940 a$=INKEY$:IF a$="" THEN 2940
2950 CLS:LOCATE 19,1:PRINT T$:PRINT:P
RINT" La torre se juega siempre en li
nea rec-ta, tanto en sentido vertical
 como hori-zontal.":PRINT"Puede captu
rar piezas enemigas de la misma forma
 que se mueve
2960 a$=INKEY$:IF a$="" THEN 2960
```

```
2970 CLS:LOCATE 19,1:PRINT AL$:PRINT:
PRINT" El alfil se mueve en diagonal,
 al igual que la torre captura de la
misma forma en que se juega.
2980 PRINT" Hay dos alfiles en cada u
no de los ban-dos : uno sobre casilla
s blancas y otro sobre negras, los cu
ales han de permane-cer sobre diagona
les del mismo color.
2990 a$=INKEY$:IF a$="" THEN 2990
3000 CLS:LOCATE 19,1:PRINT RN$:PRINT:
PRINT"  La dama es la pieza de mayor
campo de accion porque su movimento e
s una combi-nacion de los ya conocido
s de la torre y del alfil. Esta puede
 moverse en diago-nal, horizontal y v
ertical.
3010 a$=INKEY$:IF a$="" THEN 3010
3020 CLS:LOCATE 19,1:PRINT r$:PRINT:P
RINT" El rey tiene el mismo movimient
o que la dama : horizontal,vertical y
 diagonal.   Pero mientras la dama pu
ede mover las casillas que quiera el
rey solo pude ser jugado a las casill
as contiguas de una"
3030 PRINT" en una excepto cuando se
realiza el en-roque (mover a la vez e
l rey y la torre)
3040 PRINT"  El rey es el elemento fu
ndamental del juego. Hay que proteger
lo del ataque de las piezas contraria
s, al mismo tiempo, tendremos que ata
car su rey.
3050 n$="ABCDEFGH":PRINT CHR$(22);CHR
$(1):FOR i=11 TO 29 STEP 6:LOCATE i,2
0:PRINT z$:NEXT:PEN 2:LOCATE 29,20:PR
INT t$:LOCATE 20,20:PRINT r$:j=0:FOR
i=9 TO 30 STEP 3:j=j+1:LOCATE i,24:PR
INT MID$(n$,j,1):NEXT
3060 PEN 1:FOR I=1 TO 500:NEXT:a$=INK
EY$:IF a$="" THEN 3070 ELSE 3120
3070 IF SWITCH=1 THEN 3080 ELSE IF SW
ITCH=0 THEN 3090
3080 PRINT CHR$(22);CHR$(0):PEN 1:LOC
ATE 23,20:PRINT Z$:LOCATE 26,20:PRINT
 X$:PEN 2:LOCATE 8,20:PRINT T$:LOCATE
 20,20:PRINT R$:FOR I=1 TO 50:LOCATE
13,18:PRINT"ENROQUE LARGO":NEXT:GOTO
3100
3090 PRINT CHR$(22);CHR$(0):LOCATE 14
,20:PRINT X$:PEN 1:LOCATE 17,20:PRINT
 Z$:PEN 2:LOCATE 20,20:PRINT R$:PRINT
 CHR$(22);CHR$(1):LOCATE 29,20:PRINT
T$:PRINT CHR$(22);CHR$(0):FOR I=1 TO
50:LOCATE 13,18:PRINT"ENROQUE CORTO":
NEXT:GOTO 3110
3100 SWITCH=0:PRINT CHR$(22);CHR$(0):
LOCATE 20,20:PRINT X$:LOCATE 14,20:PR
INT R$:LOCATE 8,20:PRINT X$:PRINT CHR
$(22);CHR$(1):LOCATE 17,20:PRINT T$:F
OR I=1 TO 3000:NEXT:GOTO 3060
3110 SWITCH=1:PRINT CHR$(22);CHR$(0):
LOCATE 20,20:PRINT X$:LOCATE 26,20:PR
INT R$:PRINT CHR$(22);CHR$(1):PEN 1:L
OCATE 29,20:PRINT Z$:PEN 2:LOCATE 23,
20:PRINT T$:FOR I=1 TO 3000:NEXT:GOTO
 3060
3120 CLEAR INPUT:CLS:LOCATE 19,1:PRIN
T CA$:PRINT:PRINT" Ninguna pieza pued
e pasar por encima de otra excepto el
 caballo. Este salta en forma de L,es
 decir,desde la casilla que ocupa pue
de moverse a otra que diste 2 casilla
s horizontales y una vertical o
3130 PRINT"alreves,hacia atras o haci
a adelante
3140 LOCATE 16,16:PRINT z$:LOCATE 22,
16:PRINT z$:LOCATE 19,19:PRINT z$:LOC
ATE 16,22:PRINT z$:LOCATE 22,22:PRINT
 z$
3150 FOR i=1 TO 1000:NEXT:a$=INKEY$:I
F a$="" AND SWITCH=1 THEN 3160 ELSE I
F a$="" AND SWITCH=0 THEN 3170 ELSE R
ETURN
3160 PRINT CHR$(22);CHR$(1):PEN 2:SWI
TCH=0:LOCATE 16,16:PRINT ca$:FOR i=1
TO 400:NEXT:PEN 1:LOCATE 16,16:PRINT
z$:PEN 2:LOCATE 19,22:PRINT ca$:FOR i
=1 TO 400:NEXT:PRINT CHR$(22);CHR$(0)
:LOCATE 19,22:PRINT x$:GOTO 3150
3170 PRINT CHR$(22);CHR$(1):PEN 2:SWI
TCH=1:LOCATE 16,16:PRINT ca$:FOR i=1
TO 400:NEXT:PEN 1:LOCATE 16,16:PRINT
z$:PEN 2:LOCATE 22,19:PRINT ca$:FOR i
=1 TO 400:NEXT:PRINT CHR$(22);CHR$(0)
:LOCATE 22,19:PRINT x$:GOTO 3150
```





DA LA TALLA

Gigantes te espera cada martes en tu kiosco. Una revista imprescindible para todo aficionado. Gigantes del Basket te ofrece las últimas noticias, las mejores fotos, las opiniones que te interesan, las entrevistas que esperas, las anécdotas que te divierten y los los comentarios que te gusta conocer.

Pídela todas las semanas, porque Gigantes da la talla.

Trivial:
Un juego impresionante!
¿Te lo imaginas en tu ordenador?

THE COMPUTER GAME
COMMODORE-GENUS™ EDITION

SPECTRUM
COMMODORE
AMSTRAD
AMSTRAD DISK

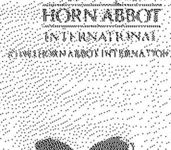
GENUS™ EDITION

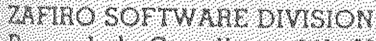
EN CASTELLANO

- VISION INSTANTANEA DE LOS RESULTADOS DE CADA JUGADOR.
- SONIDOS Y PREGUNTAS MUSICALES.
- TEMAS GRAFICOS QUE REQUIEREN MEDITACION.

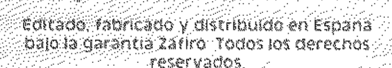


- CUESTIONES DE TEST
- 3.000 PREGUNTAS
- TIEMPO DE RESPUESTA SELECCIONABLE





ZAFIRO SOFTWARE DIVISION
Paseo de la Castellana, 141. 28046 Madrid
Tel. 459 30 04. Tel. Barna. 209 33 65. Telex: 22690 ZAFIR E



Editado, fabricado y distribuido en España bajo la garantía Zafiro. Todos los derechos reservados.

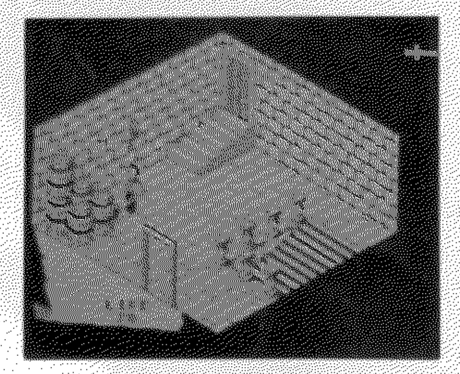


FAIRLIGHT

AMSTRAD PCW

Después de un día duro y trabajado, Ivar puso rumbo al hogar. En su mente ruda y campesina, pero inteligente, siempre estaba la misma pregunta: ¿Por qué debía trabajar de sol a sol?

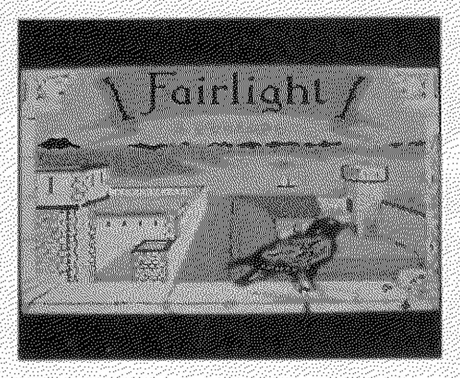
Mientras trataba de encontrar explicación a este hecho, llegó a la taberna del pueblo donde siempre se tomaba una jarra de vino mientras



hablaba con los amigos. En la puerta de la taberna vio clavado un bando y lo leyó con detenimiento. Este era su contenido:

«El señor de Cakchi busca un hombre valeroso capaz de salvar al mago del feudo y a su libro mágico que se hallan prisioneros en el castillo de Avar. Si deseas cumplir esta misión, preséntate sin demora. El que logre liberar al mago y a su libro será grandemente recompensado.»

Ivar no lo penso más y puso rumbo al castillo de Cakchi. Cuando estuvo en presencia del poderoso amo del feudo, éste le preguntó si sabía utilizar la espada, a lo que Ivar contestó que no, pero que con un poco de práctica aprendería.



Después de la audiencia, Ivar se pertrechó como correspondía a la misión que iba a emprender y a lomos de su brioso corcel se dirigió al castillo de Avar. Al llegar a la entrada del castillo se percató de que el puente levadizo estaba bajado y, muy extrañado por esto, entró hasta el patio de armas.

Lo estaba reconociendo cuando repentinamente y de la nada, vio cómo se formaba la espectral reencarnación de un guerrero. Ahora entendía el por qué de esa aparente falta de vigilancia, así como la magnitud de la tarea que iba a emprender.

Como nosotros tenemos otra por delante, y no menos importante que la de Ivar, dejaremos a éste espada en ristre y cogeremos la pluma para relatar los pormenores que nos han sucedido con **FAIRLIGHT,** juego realizado por THE EDGE.

Y para que no sucedan cosas deberemos cargar el sistema operativo, que viene con el equipo, ya que el juego no lo trae incorporado.

Después de cargado, aparecerá en pantalla un «lienzo» en el que podremos ver un castillo que se halla en las cercanías del mar.

Pulsamos una tecla, y aparecerá el menú donde se nos indicará cómo movernos, saltar, luchar, seleccionar objetos y utilizarlos, etc.

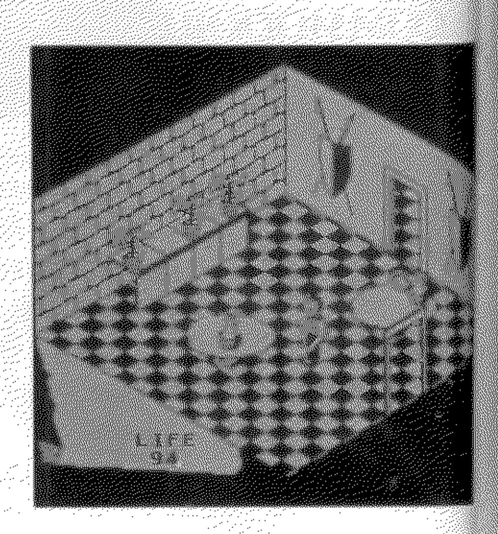
El por qué de esta explicación se halla en que también podemos usar las teclas del cursor, a nosotros nos parece menos lioso, y esto no queda muy claro en el menú.

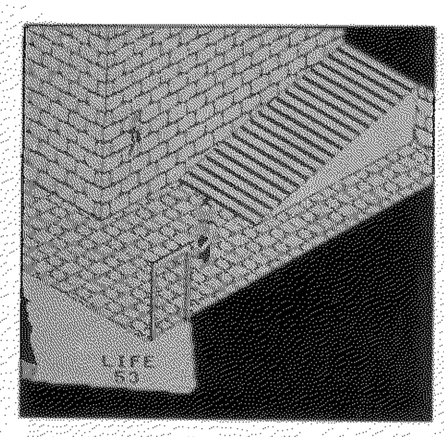
Para saltar en la dirección que queramos, deberemos mantener pulsada la tecla de la dirección en queramos ir y presionar la tecla MAYS.

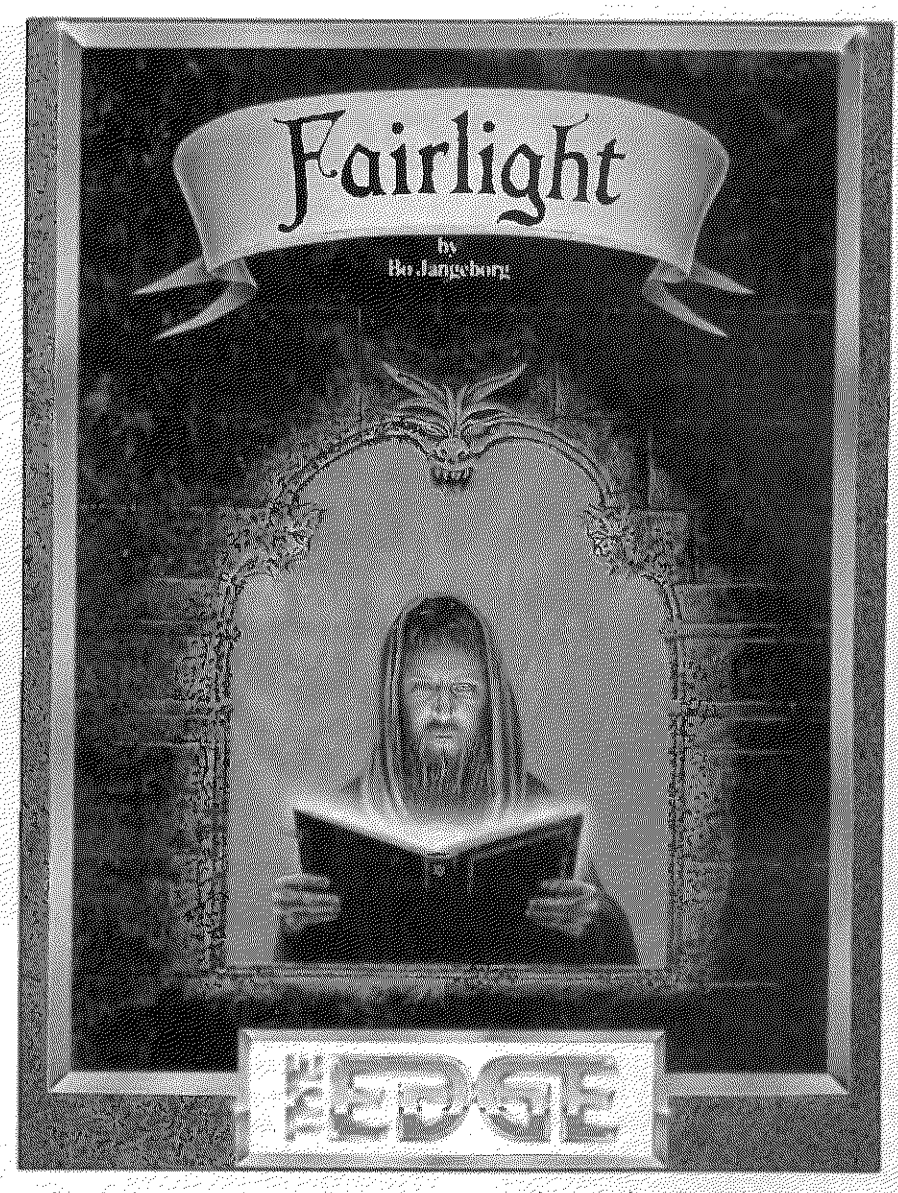
Pues una vez que ya hemos aclarado ésto o, por lo menos, lo hemos intentado, podemos seguir adelante en nuestra importante tarea de salvar al mago y a su libro.

Pero si queremos que nuestra energía vital no disminuya, tenemos que estar atentos al indicador en pantalla y procurar evitar el contacto de torbellinos, soldados reencarnadaos, gigantes barbudos y monjes negros.

Todos ellos son peligrosos, su contacto es mortal si es muy







largo, pero unos lo son más que otros. Veamos:
MONJES NEGROS. Desde el momento en que aparezca uno en pantalla no dejará de perseguirnos. Son indestructibles.
SOLDADADOS REENCARNADOS. Aparecen en pantalla desde debajo de su casco. Son duros de pelar, pero después de un arduo enfrentamiento se les puede vencer. Una vez muertos se debe recoger su casco, ya que si no, volverán a aparecer y nos atacarán de nuevo.
GIGANTES. Son grandes y poderosos. Llevan una porra con la que nos golpearán desde el momento en que nos vean. Se les puede matar con unos cuantos golpes de espada. Un **«fenómeno»** que hemos dejado para el final lo forman los TORBELLINOS, que veremos aparecer casi en cualquier pantalla. Estos voraces elementos meteorológicos destrozan todos los objetos que pillen a su paso, así como nos restarán energía si nos tocan.
Ya que hemos hablado de objetos, vamos a describir para qué sirve cada uno.
EL PAPIRO. Sirve para teletransportarnos. Se encuentra en el patio de armas del castillo, o sea en la primera pantalla.
LA CORONA. Junto con el libro mágico, sirve para terminar el juego. Se encuentra encima de la puerta inferior del patio de armas. Para llegar a alcanzar la CORONA es necesario uno de los utilísimos BARRILES que podremos encontrar en algunas de las dependencias del castillo. LLAVES, hay ocho y cada una sirve para abrir una puerta determinada.
CIRCULO ENERGETICO. Se encuentra flotando en muchos corredores y pasadizos. Su contacto nos confiere mayor velocidad, pero nos restan energía vital.
OBJETOS ALIMENTICIOS. Nos reponen energía y es interesante utilizarlos cuando nos encontremos exhaustos. Los objetos son: un pollo, un botijo, un reloj y una barra de pan.
POCION Y CRUZ. Dan paso a algunas pantallas que de otra forma sería imposible.
EL LIBRO DE LA LUZ. Una vez que lo tengamos, junto con la CORONA como ya hemos dicho, habremos terminado la misión. El LIBRO DE LA LUZ se halla situado en una remota galería subterránea del castillo.
FAIRLIGHT está confeccionado con la interesante técnica de **«FILMATION»**, por lo que vemos pasar por la pantalla de nuestro ordenador una película dentro de la que nos sentimos plenamente integrados, más aun si tenemos en cuenta que en esta ocasión el ángulo visual se ha subido, o sea, tenemos una vista más de **«pájaro».**
Gráficamente es bueno, pero pensamos que se podría mejorar la definición.
La animación está bien realizada y, tanto Ivar como todos sus oponentes se mueven con una velocidad acertada. También está muy bien logrado el movimiento de pelea que nos recuerda un poco al esgrima. Pues otro juego para PCW y, cada día son más, que pensamos está bien realizado.



PASCAL O BASIC HE AQUI EL DILEMA

Francisco G.R.

A la hora de elegir el aprendizaje de un lenguaje de programación se nos plantea la duda de cuál elegir entre los existentes, y una vez dentro de estos, cuál puede tener mayor porvenir. Nosotros vamos a intentar ver algunas de las diferencias entre los lenguajes de programación Basic y Pascal para que usted mismo pueda comparar estos dos lenguajes.

Refiriéndonos a la saga de los ordenadores **Amstrad** tenemos que decir que existen varios compiladores de Pascal y de Basic. Dentros de estos últimos tenemos que destacar el MBASIC y el CBASIC, aparte del Mallard Basic, que contienen los PCWs y el Basic Locomotive que lleva grabado en ROM los CPCs, como los compiladores o lenguajes Basic más importantes existentes en nuestro país a la venta *(hecho muy importante el que podamos disponer de ellos en caso de necesidad)*. Referente a compiladores de Pascal debemos destacar el Pascal MT-PLUS y el Pascal de Hisoft, ambos también disponibles en las listas de software españolas. Existen otros compiladores de Pascal que aún no han llegado, y esperemos que lleguen pronto como puede ser el famoso magnífico Turbo Pascal. Cada uno de los anteriores compiladores tanto para Basic como para Pascal tienen diferencias notables según la necesidad de que precisemos. Mientras el MBASIC de Microsoft está débil en la cuestión de gráficos, el CBASIC, podríamos decir que es todo lo contrario. En el Pascal de Hisoft y el Pascal-MT+ también las hay, uno opera mejor los ficheros que el otro, mientras con el MT-PLUS nos permite mayor capacidad para nuestros programas el Pascal de Hisoft se potencia con mayor facilidad de manejo de ficheros.

El lenguaje Basic prácticamente todo el mundo lo conoce aunque sólo sea de oídas, mientras el Pascal no tanto. Esto puede ser debido a que este último sea un lenguaje más científico y considerado como lenguaje de alto nivel, y el Basic por su fácil asimilación se le haya rebajado a un uso más corriente, quitándole la potencia del cual se merece. Podríamos decir refiriéndonos al lenguaje Pascal que es menos conocido, que es un lenguaje estructurado, que contiene una potencia de trabajo superior a otros lenguajes *(en cuatro palabras podríamos decir que diciendo menos dices más)*, la incorporación de procedures que le dan una gran potencia al restar del sistema los saltos a subrutinas, potenciando estas procedures que pueden incluso llamar a otras e incluso llamarse a sí misma operando con las variables que contiene dentro de sí. También podemos decir que está dotado de procedimientos recursivos, tal y como nos habíamos referido anteriormente al hecho de cambiar las variables y llamarse luego a sí mismo.

Diferencias en la práctica y similitud

Hemos creado dos programas uno que nos pasa los grados centígrados a grados Fahrenheit, y un segundo que nos representa unas curvas senoidales y cosenoidales.

El primero es el paso de grados de una escala a otra. Vamos a ver el listado escrito en Pascal y a continuación escrito en Basic:

Programa 'CALCULO'

Función: Calcula grados centígrados a grados Fahrenheit.

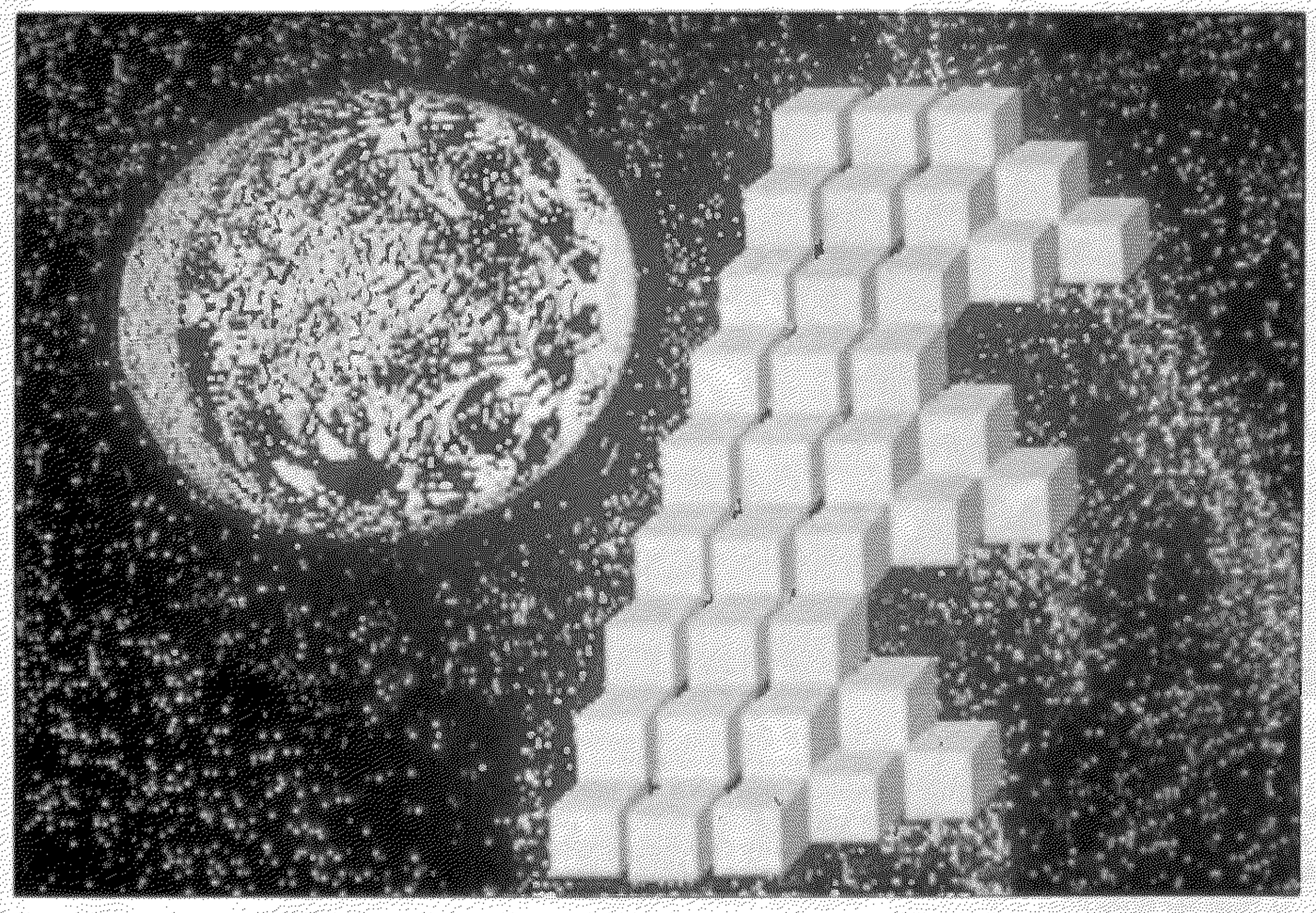
Realización: Pascal.

Listado:

```
PROGRAMA CALCULO (INPUT,
OUTPUT)
  CONST CONVERSION=1.8;
  DESPLA=32.0;
  VAR CENTEMP:INTEGER;
  FAHRTEMP:REAL;
  ( *Comienzo del cuerpo del
Programa principal* )
  BEGIN
  WRITELN('CENTIGRADOS
FAHRENHEIT');
  READ(CENTEMP);
  FAHRTEMP:=CENTEMP*CON-
VERSION
+DESPLA;
  WRITE(CENTEMP:2,' ':9,FAHR-
TEMP:7:1)
  END
```

Descripción del programa.

Lo primero que designamos es el nombre del programa que denominaremos 'CALCULO', el cual está definido como un fichero



de entrada y salida estándar en la máquina que dispongamos. Esto se refiere que los datos entrarán por el teclado y saldrán por la pantalla. Seguidamente declaramos las variables que vamos a necesitar en la realización de nuestro programa:

— CONVERSION
— DESPLA
— CENTEMP
— FARHRTEMP

Siendo las dos primeras variables constantes, la tercera es una variable entera y la última la definimos como una variable real. Una vez definidas las variables comenzaremos con la creación del programa en la cual la orden 'BEGIN' es la encargada del comienzo del mismo, para que el sistema pueda reconocer la entrada. La segunda función escrita en un 'WRITELN', que es la equivalente como después comprobaremos a un 'PRINT' a secas. Luego tenemos una nueva instrucción dentro de nuestro programa, se trata de 'READ', esta instrucción se puede definir como un 'INPUT' en Basic. A continuación tenemos una serie de cálculos que bien podríamos traducir por un 'LET' con un valor tras el resultado de unas operaciones. Para finalizar tenemos que nos escribe en la pantalla el resultado obtenido del paso de una unidad a otra, y el final de la ejecución.

Como hemos podido comprobar hay ciertos símbolos que forman parte del léxico del Pascal, estos símbolos son, por ejemplo, los puntos y comas que aparecen al final de una línea, con lo cual indica que esa línea de programa ha terminado, cosa que en Basic se denota cuando se encuentra la numeración de la siguiente línea. Pero al revés que en el Basic es obligatorio colocar un número para ver que efectivamente se ha acabado la anterior, en Pascal hay varios casos en los cuales no se denota el punto y coma. Estos casos pueden al principio del programa, después de un 'BEGIN', antes de producirse un 'END', etc. Otro símbolo particular es el punto que insertamos después del 'END.', éste lo indicamos para expresar que esta es la última línea del programa.

También podemos observar que existe una línea que está encerrada entre paréntesis, esto nos indica que lo que a continuación sigue es un comentario que el ordenador no lo tiene encuenta a la hora de correr el programa, sino que más bien es una ayuda al programador para localizar las partes de su programa. Estas líneas símiles en Basic con el 'REM', van precedidas por un paréntesis y a continuación un '*'.

A continuación tenemos el mismo programa realizado en Basic para poder observar mejor las diferencias:

Programa 'CAMBIO'

Función: Calcula grados centígrados a grados Fahrenheit.

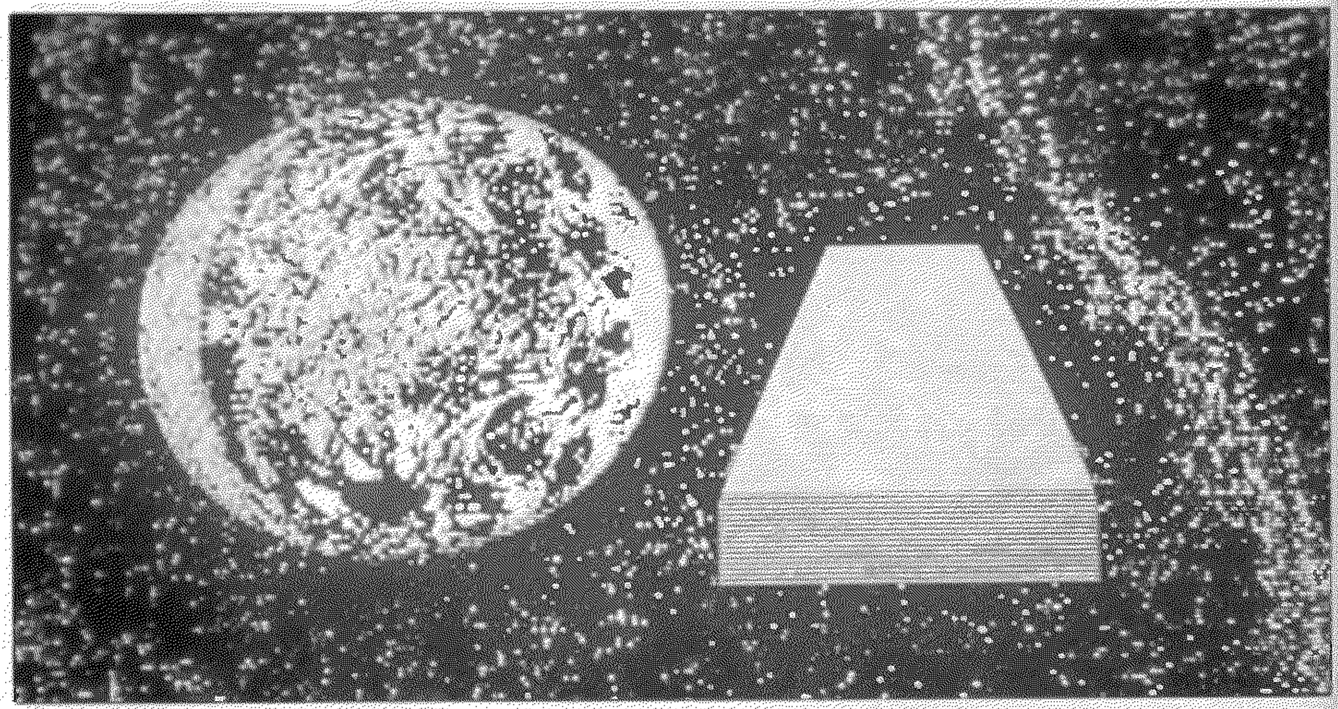
Realización: Basic.

Listado:

```
  10 LET CONVERSION=1.8
  20 LET DESPLA=32.0
  30 INPUT CENTEMP
  40 LET FAHRTEMP=CENTEMP*
CONVERSION+DESPLA
  50 PRINT "CENTIGRADOS
FAHRENHEIT"
  60 PRINT
CENTEMP;"
FAHRTEMP
  70 END
```

Descripción del programa.

La descripción del programa se



ha realizado anteriormente, pero debemos dejar constancia que no era necesario el tener que escribir el comando 'LET', pero en Basics diferentes al de los CPC de **Amstrad,** hay que denotarlo.

El segundo programa que vamos a ver se trata de realizar la representación en pantalla de unas curvas senoidales y cosenoidales:

Programa 'CURVAS'

Función: Representación de una función.

Realización: Pascal.

Listado:

```
PROGRAM
CURVAS(INPUT,OUTPUT)
 CONST H= 34
          C= 6.2831; (* 2 *
       PI*)
 VAR X,Y,D :REAL;
      I,N,S,LIM:INTEGER
 BEGIN
 READ(D,S,LIM);
 FOR I:0 TO LIM DO
 BEGIN
 X:=D*1;
 Y:=EXP(—X)*SIN(C*X);
 N:=ROUND(S*Y)+H;
 WHILE N=0 DO
 BEGIN
 WRITE('      ');
 N:=N—1
 END;
 WRITELN('*')
 END
 END.
```

Descripción del programa.
Como siempre denotamos el nombre del programa y el tipo de programa del cual se va a tratar, en este caso será de nuevo un programa de salida por pantalla y entrada de datos por teclado. A continuación definimos las variables siempre comenzando por las variables constantes y seguidamente las reales y enteras:

— **H** Variable constante
— **C** Variable constante
— **X** Variable real
— **Y** Variable real
— **D** Variable real
— **I** Variable entera
— **N** Variable entera
— **S** Variable entera
— **LIM** Variable entera

Una vez definidas todas ellas, comenzaremos con la estructura del programa. Este comenzará con un 'BEGIN', a continuación le precerá un 'READ' (entrada de dato por el teclado), y se iniciará un bucle que irá desde 0 hasta la variable 'LIM'. Luego tendremos el comienzo de una subrutina que también podíamos haber puesto en forma de procedure, en esta parte nueva del programa vamos calculando los valores para que más tarde se representen gráficamente en la pantalla. Luego tenemos la parte en la cual se encarga de representar la curva, la cual debe de ir mediante 'WRITELN' *(instrucción igual que un Print en Basic, imprime y se pasa a la siguien línea)* o 'WRITE' *(ésta la podríamos igualar como un Print seguido de un punto y coma, o sea, que el siguiente carácter a imprimir se coloque a continuación del anterior escrito).* Y finalizaremos con el 'END.''

Ahora lo vamos a desarrollar en Basic:

Programa 'GRAFICO'

Función: Representación de una función.

Realización: Basic.

Listado:

```
 10 LET H=34
 20 LET C=6.2831:REM 2*PI
 30 INPUT D,S,LIM
 40 FOR I=0 TO LIM
 50 X=D*I
 60 Y=EXP(—X)*SIN(C*X)
 70 N=ROUND(S*Y)+H
 80 WHILE N=0
 90 PRINT "      "
100 LET N=N+1
110 WEND
120 PRINT "*"
130 NEXT I
140 END
```

Como podemos comprobar la estructura de uno y otro son prácticamente casi idénticas, lo hemos intentado realizar de una manera idéntica para que el lector pueda comprobar la similitud entre los diferentes comandos y la nomenclatura a seguir. La diferencia grande de uno a otro la podríamos tener y de hecho la hay en el desarrollo y cálculo de una a la otra, siendo el programa compilado en Pascal más rápido que compilado en Basic.

CUCAL

¡Maldita cucaracha! Un arcade con una buena presentación y en el que nuestra habilidad y reflejos serán indispensables para conseguir el éxito.

Al comenzar el juego aparecen las instrucciones y teclas de uso. El juego consta de diez fases distintas, las cuales, aumentan en dificultad a medida que vamos pasándolas. En cada fase hay tres frutas (excepto en la fase diez que hay tres llaves), las cuales nos dan la posibilidad de pasar a una fase superior después de haber sido comidas.

En las fases 4, 6, y 8 podrán ver una bombillita, la cual, no se puede coger aparentemente, pero se darán cuenta que al comer las tres manzanas que hay en estas fases se abre la puerta de salida y, además, desaparece una mina de la pantalla, la cual, nos deja la posibilidad de coger la bombilla que, por supuesto no es obligatoria.

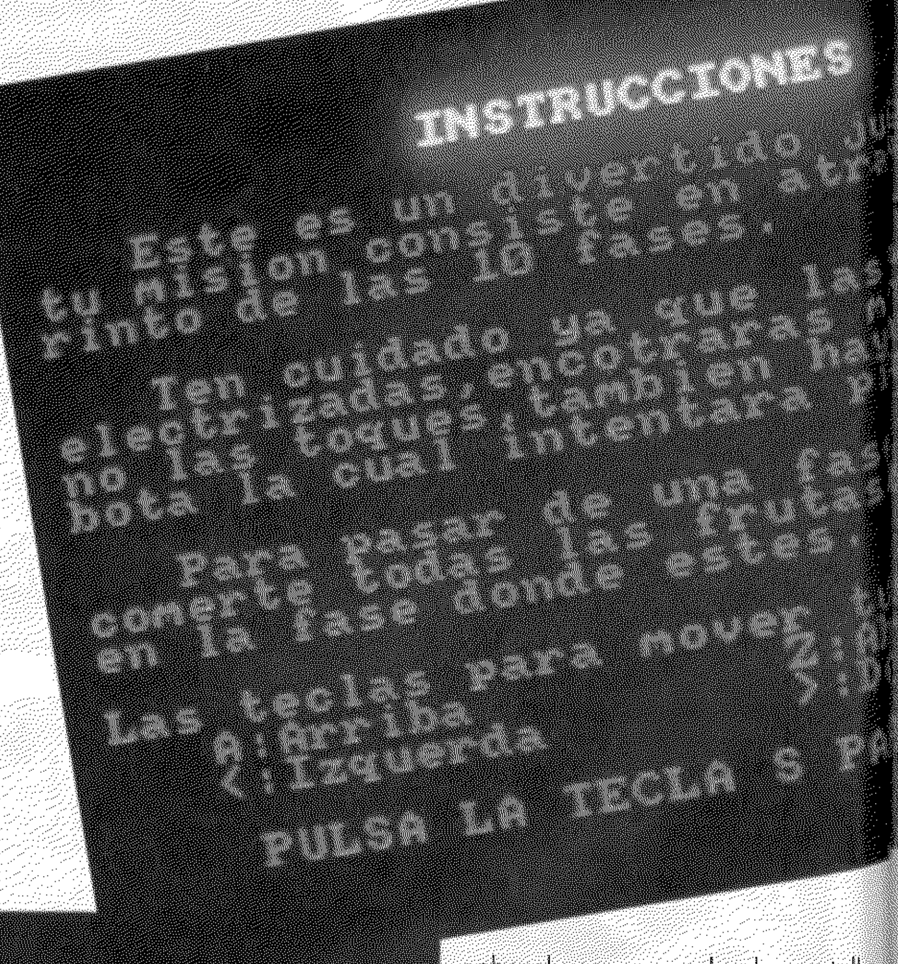
Y se preguntarán: ¿Para qué sirve esto de las bombillas?, pues sirve para que al llegar a la fase nº 9 (fase oscura) tengamos luz, es decir, si durante nuestro recorrido hemos cogido las tres bombillas en la fase nº 9 podremos ver la situación de las minas, de lo contrario, las minas no aparecerán, pero se encuentra en su lugar.

Después de haber cogido las tres llaves de la fase nº 10 y haber salido verán cómo aparece un letrero de enhorabuena, se rodea la pantalla de minas y después de haber sonado cuatro veces la música y de haber pasado otras cuatro veces dos cucarachas aparecerá un letrero que nos dice que pulsemos la letra a para jugar otra vez.

Si, por el contrario, nuestra misión fracasa y nos matan las tres vidas, veremos como un final, una bota aplastando a una cucaracha y un letrero que nos dice que pulsemos la s para jugar otra vez.

En cada fase y para dificultar la misión, habrá una bota que intentará pisarnos.

Para que vosotros podáis ver todas las fases sin que os moleste la bota, aquí os doy el remedio.





Añadid la línea:
145 u=1: q=1
Estas variables dan las coordenadas a la bota, por lo tanto, veréis que la bota aparece en la esquina superior izquierda del monitor.

Las vidas que quedan se representan por un cucaracha y cuando te matan, la cucaracha se convierte en un esqueleto.

El juego tiene un marcador que va acumulando la puntuación obtenida. Todos los movimientos, choques, etc., van acompañados de efectos especiales.

VARIABLES PRINCIPALES

u,q: Coordenadas de la bota.
a,d: Coordenadas de la cucaracha.
li: Vidas.
e(x): Variable para la localización de la cucaracha.
man: Frutas comidas.
pan: nº de fase.
j,k: Coordenadas minas.
i,s: Coordenadas bombillas.

SUBRUTINAS PRINCIPALES:

150-270: Programa principal.
680-740: Comprueba si ha chocado o comido.
990-1190: Coloca minas, llaves y bombillas.
1200-1790: Pantallas.

Serie ORO

```
10 '**************
20 '*            *
30 '* ESCARABAJO *
40 '*    POR     *
50 '* A. FRIAS   *
60 '*            *
70 '**************
80 GOSUB 490
90 MODE 1:BORDER 0:INK 0,0:GOSUB 197
0
100 CLS:CLEAR:a=11:d=9:m=14:li=3:u=1
:q=1:DIM e(120)
110 GOSUB 960:GOSUB 920:GOSUB 760:GO
SUB 1210
120 m1$=q3$
130 m2$=q4$
140 RESTORE
145 u=1:q=1
150 FOR n=1 TO 15
160 IF INKEY(69)=0 THEN GOSUB 290
170 IF INKEY(71)=0 THEN GOSUB 340
180 IF INKEY(39)=0 THEN GOSUB 390
190 IF INKEY(31)=0 THEN GOSUB 440
200  LOCATE u,q:PRINT"  ":LOCATE u,q
+1:PRINT"  ":IF u=a AND d=q THEN SOU
ND 3,800,30,15:GOSUB 610
210 READ u,q
220 SOUND 1,400,0.7,15
230  LOCATE u,q:PRINT q15$:LOCATE u,
q+1:PRINT q16$
240 IF u=a AND d=q THEN SOUND 3,800,
30,15:GOSUB 610
250 NEXT n
260 nki=nki+1:IF nki/2=nki\2 THEN RE
STORE
270 GOTO 150
280 '*****ARRIBA*****
290 LOCATE a,d:PRINT"  ":LOCATE a,d+
1:PRINT"  "
300 m=m-12:d=d-2
```

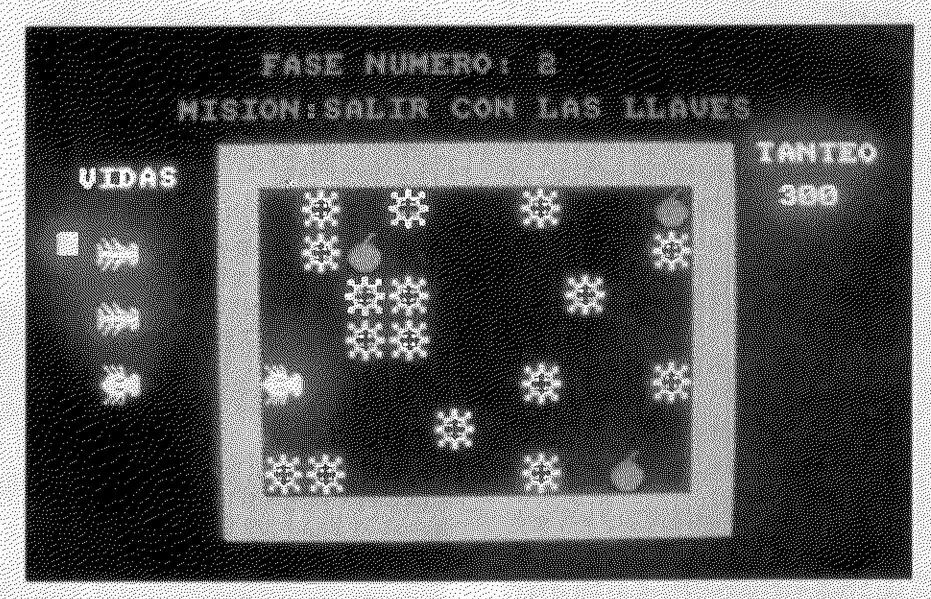


```
310 GOSUB 690
320 RETURN
330 '*****ABAJO*****
340 LOCATE a,d:PRINT"  ":LOCATE a,d+
1:PRINT"  "
350 m=m+12:d=d+2
360 GOSUB 690
370 RETURN
380 '*****IZQUIERDA*****
390 LOCATE a,d:PRINT"  ":LOCATE a,d+
1:PRINT"  ":m1$=q5$:m2$=q6$
400 m=m-1:a=a-2
410 GOSUB 690
420 RETURN
430 '******DERECHA******
440 LOCATE a,d:PRINT"  ":LOCATE a,d+
1:PRINT"  ":m1$=q3$:m2$=q4$
450 m=m+1:a=a+2
460 GOSUB 690
470 RETURN
480 '**********caracteres***********
*
490 SYMBOL AFTER 200
500 SYMBOL 241,0,36,18,9,15,31,60,12
7:SYMBOL 242,0,128,64,64,198,239,61,
255:SYMBOL 243,255,61,239,198,64,64,
128:SYMBOL 244,127,60,31,15,9,18,36
510 SYMBOL 245,0,128,140,140,112,16,
30,14:SYMBOL 246,14,80,16,112,140,14
0,128:SYMBOL 247,0,1,49,49,14,8,10,1
12:SYMBOL 248,112,10,8,14,49,49,1
520 SYMBOL 249,0,0,252,252,252,252,2
52,252:SYMBOL 250,252,254,254,254,25
4,254,252:SYMBOL 251,1,26,59,127,127
,127,63
530 SYMBOL 201,0,0,0,0,1,7,15,31:SYM
BOL 202,0,48,96,192,128,224,240,248:
SYMBOL 203,63,63,63,63,31,15,7:SYMBO
L 204,252,252,252,252,248,240,224
540 SYMBOL 205,0,0,34,17,153,205,68,
255:SYMBOL 206,255,68,205,153,17,34:
SYMBOL 209,0,0,0,0,134,231,89,255:SY
MBOL 208,255,89,231,134
550 SYMBOL 210,0,1,2,2,99,247,188,25
5:SYMBOL 211,255,188,247,99,2,2,1:SY
MBOL 212,0,36,72,144,240,248,60,254:
SYMBOL 213,254,60,248,240,144,72,36
560 SYMBOL 214,0,0,0,0,0,31,63:SYM
BOL 215,0,0,0,0,0,28,190:SYMBOL 21
6,127,255,255,255,102,66,132,198:SYM
BOL 217,243,255,254,240,48,32,64,96
570 SYMBOL 218,0,0,7,24,32,67,128,12
8:SYMBOL 219,128,128,67,32,24,7:SYMB
OL 220,0,0,128,64,32,30,199,63:SYMBO
L 221,63,199,30,32,64,128:SYMBOL 222
,252,252,252,252,252,252,252,252
580 SYMBOL 223,0,12,18,33,225,225,18
,12:SYMBOL 224,0,0,108,108,111,127
590 RETURN
600 '**********PIERDE VIDA**********
*
610 li=li-1:PO=PO+3
620 BORDER 0,15
630 LOCATE 3,8+PO:PRINT q7$:LOCATE 3
,9+PO:PRINT q8$
640 FOR i=1  TC 2000 :NEXT i : BORDE
R 0
650 IF li=0 THEN GOSUB 840
660  LOCATE u,q:PRINT"  ":LOCATE u,q
+1:PRINT"  "
670 a=11:d=9:m=14:u=1:q=1:LOCATE a,d
:PRINT q3$:LOCATE a,d+1:PRINT q4$:GO
TO 120
680 '**********COMPRUEBA CHOQUE*****
*****
```

FASE NUMERO: 1
MISION:SALIR CON LAS LLAVES
TANTEO
VIDAS

Serie ORO

```
690 IF man>=3 AND m=96 THEN IF man>3 THEN uno=uno+1:GOSUB 820 ELSE GOSUB 820
700 IF e(m)=0 THEN SOUND 3,2500,30,15,10,9,15:GOSUB 610
710 IF e(m)=2 THEN SOUND 1,40,4,15:TA=TA+100:LOCATE 34,9:PRINT TA:man=man+1:e(m)=1:IF man=3 THEN LOCATE 31,21:PRINT"  ":LOCATE 31,22:PRINT"  ":IF ro=4 THEN GOSUB 1950
720 SOUND 1,200,4,15
730 LOCATE a,d:PRINT m1$:LOCATE a,d+1:PRINT m2$
740 RETURN
750 ' **********MUESTRA VIDAS*********
760 LOCATE 2,8:PRINT"VIDAS":l=8
770 FOR N=1 TO 3
780 l=l+3:LOCATE 3,l:PRINT q3$:LOCATE 3,l+1::PRINT q4$
790 NEXT
800 RETURN
810 '*******cambio de pantalla********
820 FOR K=1500 TO 300 STEP -8:SOUND 1,K,0.5,15:NEXT
830 ON ida GOTO 1260,1310,1360,1420,1470,1520,1570,1630,1750,2180
840 '*********final***********
850 MODE 1:BORDER 14
860 CLEAR:GOSUB 960:LOCATE 19,20:PRINT q13$:LOCATE 19,21:PRINT q14$:LOCATE 1,12:PRINT"   Ohhhhhh!!!!!!!   LO SIENTO!!!!!!!!"
870 PEN 3:FOR x=1 TO 20:LOCATE 19,x:PRINT q17$:LOCATE 19,x+1:PRINT q16$:SOUND 1,ABS(x-20)*20,3,15:FOR r=1 TO 10:NEXT r:NEXT x
880 FOR n=1 TO 1000:NEXT
890 LOCATE 6,23:PRINT"PULSA S PARA JUGAR OTRA VEZ"
900 A$=INKEY$:IF A$="S" OR A$="s" THEN RUN ELSE GOTO 900
910 '*********dibuja pantalla**********
920 PEN 1:FOR n=9 TO 32:LOCATE n,7:PRINT CHR$(207):LOCATE n,8:PRINT CHR$(207):LOCATE n,23:PRINT CHR$(207):LOCATE n,24:PRINT CHR$(207):NEXT
930  FOR n=9 TO 22:LOCATE 9,n:PRINT q18$:LOCATE 31,n:PRINT q18$:NEXT
940 RETURN
950 '*******variables de caracteres*******
960 q1$=CHR$(218)+CHR$(220):q2$=CHR$(219)+CHR$(221):q3$=CHR$(241)+CHR$(242):q4$=CHR$(244)+CHR$(243):q5$=CHR$(210)+CHR$(212):q6$=CHR$(211)+CHR$(213):q7$=CHR$(205)+CHR$(209):q8$=CHR$(206)+CHR$(208):q9$=CHR$(201)+CHR$(202):q10$=CHR$(203)+CHR$(204)
970 q11$=CHR$(247)+CHR$(245):q12$=CHR$(248)+CHR$(246):q13$=CHR$(214)+CHR$(215):q14$=CHR$(216)+CHR$(217):q15$=" "+CHR$(249):q16$=CHR$(251)+CHR$(250):q17$=" "+CHR$(222):q18$=CHR$(207)+CHR$(207)
980 RETURN
990 '********subrutina de pantalla*********
1000 FOR i=1 TO 110 :e(i)=1:NEXT
1010 FOR i=1 TO 108 STEP 12:e(i)=0:e(i-1)=0:NEXT
1020 FOR i=1 TO 12:e(i)=0:e(i+96)=0:NEXT
1030 PEN 1:LOCATE 31,21:PRINT q18$:LOCATE 31,22:PRINT q18$
1040 FOR j=1 TO 14:LOCATE 11,8+j:PRINT"                    ":NEXT j
```




```
1050 RESTORE:FOR n=1 TO dal:READ o,p:NEXT
1060 FOR e=1 TO du:READ h,p
1070 LOCATE h,p:PRINT q11$:LOCATE h,p+1:PRINT q12$
1080 NEXT e
1090 PEN 3
1100 FOR n=1 TO 3
1110 READ j,k:LOCATE j,k:PRINT q9$:LOCATE j,k+1:PRINT q10$
1120 NEXT
1130 LOCATE 11,3:PRINT"FASE NUMERO:";pan
1140 LOCATE 7,5:PRINT"MISION:SALIR CON LAS LLAVES "q$
1150 PEN 2:LOCATE a,d:PRINT q3$:LOCATE a,d+1:PRINT q4$
1160 LOCATE 34,7:PRINT"TANTEO"
1170 man=0:q=1:u=1
1180 IF ro=4 THEN READ i,s:LOCATE i,s:PRINT q1$:LOCATE i,s+1:PRINT q2$
1190 RETURN
1200 '************primera pantalla*********
1210 PAN=1:ida=1:dal=30:du=15
1220 GOSUB 1000
1230 e(18)=0:e(27)=0:e(33)=0:e(40)=0:e(41)=0:e(45)=0:e(52)=0:e(53)=0:e(59)=0:e(64)=0:e(67)=0:e(70)=0:e(88)=0:e(91)=0:e(79)=0:e(23)=2:e(43)=2:e(86)=2
1240 RETURN
1250 '**************segunda pantalla************
1260 a=11:d=9:m=14:man=0:ida=2:pan=2:dal=47:du=17
1270 GOSUB 1000
1280 e(15)=0:e(20)=0:e(27)=0:e(35)=0:e(40)=0:e(41)=0:e(45)=0:e(52)=0:e(53)=0:e(68)=0:e(71)=0:e(78)=0:e(86)=0:e(87)=0:e(94)=0:e(17)=0:e(23)=2:e(28)=2:e(94)=2
1290 GOTO 120
1300 '***********tercera pantalla***********
1310 pan=3:ida=3:dal=68:du=21:a=11:d=9:m=14:man=0
1320 GOSUB 1000
1330 e(15)=0:e(18)=0:e(23)=0:e(27)=0:e(28)=0:e(33)=0:e(39)=0:e(41)=0:e(45)=0:e(53)=0:e(70)=0:e(71)=0:e(74)=0
```

```
:e(75)=0:e(78)=0:e(81)=0:e(86)=0:e(8
7)=0:e(89)=0:e(90)=0:e(91)=0:e(40)=2
:e(17)=2:e(83)=2
1340 GOTO 120
1350 '***********cuarta pantalla****
*******
1360 pan=4:ida=4:dal=91:du=26:a=11:d
=9:m=14:man=0:ro=4:x1=15:y1=21:x2=88

1370 GOSUB 1000
1380 e(18)=0:e(20)=0:e(22)=0:e(23)=0
:e(27)=0:e(28)=0:e(35)=0:e(45)=0:e(5
0)=0:e(52)=0:e(57)=0:e(59)=0:e(62)=0
:e(67)=0:e(70)=0:e(71)=0:e(74)=0:e(7
5)=0
1390 e(78)=0:e(81)=0:e(86)=0:e(87)=2
:e(90)=0:e(91)=0:e(88)=0:e(19)=0:e(4
6)=2:e(83)=2:e(89)=2
1400 GOTO 120
1410 '**********quinta pantalla*****
*******
1420 pan=5:ida=5:dal=121:du=23:a=11:
d=9:m=14:man=0:ro=0
1430 GOSUB 1000
1440 e(15)=0:e(18)=0:e(22)=0:e(23)=0
:e(27)=0:e(41)=0:e(43)=0:e(45)=0:e(5
0)=0:e(59)=0:e(62)=0:e(64)=0:e(66)=0
:e(68)=0:e(70)=0:e(71)=0:e(75)=0:e(7
8)=0:e(79)=0:e(83)=0:e(89)=0:e(90)=0
:e(91)=0:e(17)=2:e(74)=2:e(92)=2
1450 GOTO 120
1460 '************sexta pantalla****
*************
1470 pan=6:ida=6:dal=147:a=11:d=9:ma
n=0:ro=4:du=20:x1=11:y1=19:x2=74:m=1
4
1480 GOSUB 1000
1490 e(18)=0:e(22)=0:e(27)=0:e(28)=0
:e(33)=0:e(39)=0:e(41)=0:e(46)=0:e(5
0)=0:e(66)=0:e(67)=0:e(70)=0:e(71)=0
:e(74)=0:e(75)=0:e(77)=0:e(87)=0:e(8
8)=0:e(91)=0:e(92)=0:e(23)=2:e(62)=2
:e(89)=2:e(86)=2
1500 GOTO 120
1510 '**********septima pantalla****
*******
1520 pan=7:ida=7:dal=171:a=11:d=9:ma
n=0:ro=0:du=22:m=14
1530 GOSUB 1000
1540 e(19)=0:e(23)=0:e(27)=0:e(28)=0
:e(33)=0:e(39)=0:e(41)=0:e(43)=0:e(4
5)=0:e(46)=0:e(53)=0:e(57)=0:e(64)=0
:e(67)=0:e(68)=0:e(70)=0:e(71)=0:e(7
4)=0:e(75)=0:e(79)=0:e(94)=0:e(40)=2
:e(59)=2:e(86)=2
1550 GOTO 120
1560 '***********octava pantalla****
****
1570 pan=8:ida=8:dal=196:a=11:d=9:ma
n=0:ro=4:du=24:m=14:x1=11:y1=17:x2=6
2
1580 GOSUB 1000
1590 e(15)=0:e(18)=0:e(22)=0:e(27)=0
:e(28)=0:e(33)=0:e(39)=0:e(41)=0:e(4
2)=0:e(43)=0:e(46)=0:e(53)=0:e(57)=0
:e(62)=0:e(66)=0:e(67)=0:e(71)=0:e(7
5)=0:e(79)=0:e(86)=0
1600 e(87)=0:e(88)=0:e(89)=0:e(94)=0
:e(17)=2:e(40)=2:e(23)=2:e(74)=2
1610 GOTO 120
1620 '************novena pantalla***
******
1630 pan=9:ida=9:dal=224:a=11:d=9:ma
n=0:ro=0:du=25:m=14
1640 FOR j=1 TO 14:LOCATE 11,8+j:PRI
NT"                ":NEXT j
1650 LOCATE 15,12:PRINT"FASE OSCURA"
1660 LOCATE 14,14:PRINT" BOMBILLAS:
";UNO
1670 IF uno=3 THEN LOCATE 17,17:PRIN
T"HAY LUZ":GOTO 1690
1680 LOCATE 16,17:PRINT"NO HAY LUZ":
q11$=" ":q12$=" "
1690 FOR j=1 TO 2500:NEXT
1700 GOSUB 1000
1710 e(18)=0:e(19)=0:e(22)=0:e(27)=0
:e(28)=0:e(33)=0:e(40)=0:e(42)=0:e(4
3)=0:e(44)=0:e(46)=0:e(62)=0:e(66)=0
:e(67)=0:e(71)=0:e(74)=0:e(75)=0:e(7
8)=0:e(79)=0:e(86)=0:e(87)=0:e(88)=0
:e(89)=0
1720 e(90)=0:e(91)=0:e(94)=0:e(20)=2
:e(23)=2:e(89)=2
1730 GOTO 120
1740 '**********decima pantalla*****
***
1750 GOSUB 960
1760 pan=10:ida=10:dal=252:a=11:d=9:
man=0:ro=0:du=28:m=14:q9$=" ":q10$=
CHR$(224)+CHR$(223)
1770 GOSUB 1000
1780 e(15)=0:e(17)=0:e(19)=0:e(20)=0
:e(27)=0:e(33)=0:e(35)=0:e(39)=0:e(4
1)=0:e(42)=0:e(44)=0:e(45)=0:e(53)=0
:e(57)=0:e(59)=0:e(62)=0:e(64)=0:e(6
7)=0:e(71)=0:e(75)=0
1790 e(77)=0:e(79)=0:e(81)=0:e(86)=0
:e(91)=0:e(92)=0:e(94)=0:e(83)=2:e(2
3)=2:e(87)=2
1800 GOTO 120
1810 '***********datas************

1820 DATA 29,21,27,19,25,21,23,19,25
,17,23,15,21,15,19,15,17,17,15,19,13
,17,13,15,11,13,11,11,11,9
1830 DATA 29,21,29,21,27,19,25,21,23
,19,25,17,27,15,29,13,27,11,25,9,23,
11,21,11,19,11,17,11,15,9
1840 DATA 19,9,13,11,25,11,15,13,17,
13,25,13,15,15,17,15,29,15,15,17,21,
17,27,17,21,19,15,21,21,21,29,9,21,1
3,11,21
1850 DATA 13,9,17,9,23,9,13,11,29,11
,15,13,17,13,25,13,15,15,17,15,23,17
,29,17,19,19,11,21,13,21,23,21,27,21
,29,9,15,11,15,21
1860 DATA 13,9,19,9,29,9,13,11,15,11
,25,11,13,13,17,13,25,13,17,15,27,17
,29,17,11,19,13,19,19,19,25,19,11,21
,13,21,17,21,19,21,21,21,17,9,15,13,
29,19
1870 DATA 19,9,21,9,23,9,27,9,29,9,1
3,11,15,11,29,11,25,13,11,15,15,15,2
5,15,29,15,11,17,21,17,27,17,29,17,1
1,19,13,19,19,19,25,19,11,21,15,21,1
9,21,21,21,17,21,27,13,29,19,13,21
1880 DATA 13,9,19,9,27,9,29,9,13,11,
17,13,21,13,25,13,11,15,29,15,11,17,
15,17,19,17,23,17,27,17,29,17,13,19,
19,19,21,19,29,19,17,21,19,21,21,21,
11,19,17,9,23,21
1890 DATA 19,9,27,9,13,11,15,11,25,1
1,13,13,17,13,27,13,11,15,19,17,21,1
7,27,17,29,17,11,19,13,19,17,19,13,2
1,15,21,21,21,23,21,29,9,11,17,17,21
,11,21
```



```
1900 DATA 21,9,29,9,13,11,15,11,25,1
1,13,13,17,13,21,13,25,13,27,13,17,1
5,25,15,15,17,21,17,23,17,27,17,29,1
7,11,19,13,19,21,19,17,21,27,21,15,1
3,29,15,11,21
1910 DATA 13,9,19,9,27,9,13,11,15,11
,25,11,13,13,17,13,19,13,21,13,27,13
,17,15,25,15,11,17,19,17,21,17,29,17
,13,19,21,19,11,21,13,21,15,21,17,21
,27,21,17,9,29,9,15,13,11,19
1920 DATA 19,9,21,9,27,9,13,11,15,11
,25,11,15,13,19,13,21,13,23,13,27,13
,11,17,19,17,21,17,29,17,11,19,13,19
,19,19,21,19,11,21,13,21,15,21,19,21
,21,21,27,21,23,9,29,9,17,21
1930 DATA 13,9,17,9,19,9,21,9,23,9,1
3,11,25,11,29,11,13,13,17,13,19,13,2
3,13,25,13,17,15,25,15,29,15,11,17,1
5,17,21,17,29,17,13,19,17,19,21,19,2
5,19,11,21,21,21,23,21,27,21,29,9,13
,21,29,19,11,19
1940 '********abre puertas de bombil
las******
1950 LOCATE x1,y1:PRINT"  ":LOCATE x
1,y1+1:PRINT"  ":e(x2)=1
1960 RETURN
1970 ' *******instrucciones*******
1980 PEN 2:LOCATE 13,3:PRINT"INSTRUC
CIONES":PEN 3
1990 PRINT:PRINT"   Este es un diver
tido juego en el cualtu mision consi
ste en atravesar el labe-rinto de la
s 10 fases."
2000 PRINT:PRINT"   Ten cuidado ya q
ue las paredes estan electrizadas,en
cotraras minas explosivasno las toqu
es,tambien hay una molesta   bota la
 cual intentara pisarte."
2010 PRINT:PRINT"   Para pasar de un
a fase a otra debes  comerte todas l
as frutas que encuentres en la fase
donde estes."
2020 PRINT:PRINT"Las teclas para mov
er tu escarabajo:        A:Arriba
     Z:Abajo                <:Izquer
da        >:Derecha"
2030 PRINT:PRINT"     PULSA LA TECLA
S PARA JUGAR"
2040 GOSUB 2070
2050 GOTO 2040
2060 '*******musica*********
2070 SOUND 1,478,100,6
2080 SOUND 2,379,50,0:SOUND 2,379,50
,6
2090 SOUND 4,319,75,0:SOUND 4,319,25
,6
2100 SOUND 1,506,100,7
2110 SOUND 2,426,25,0:SOUND 2,426,75
,6
2120 SOUND 4,319,50,0:SOUND 4,319,50
,6
2130 SOUND 1,568,300,6
2140 SOUND 4,284,100,0:SOUND 4,284,2
5,6:SOUND 4,239,50,6:SOUND 4,190,125
,6
2150 FOR b=1 TO 800:A$=INKEY$:IF A$=
"s" OR A$="S" THEN GOTO 100 ELSE NEX
T b
2160 RETURN
2170 '*********terminar*********
2180 CLS:CLEAR:GOSUB 960:PEN 1:FOR n
=1 TO 40 STEP 2:LOCATE n,2:PRINT q11
$:LOCATE n,3:PRINT q12$:LOCATE n,22:
PRINT q11$:LOCATE n,23:PRINT q12$:NE
XT
2190 FOR n=1 TO 22 STEP 2:LOCATE 1,n
+1:PRINT q11$:LOCATE 1,n+2:PRINT q12
$:LOCATE 39,n+1:PRINT q11$:LOCATE 39
,n+2:PRINT q12$:NEXT
2200 PEN 3:LOCATE 3,13:PRINT" ENHORA
BUENA!!!!  LO CONSEGUISTE!!!"
2210 FOR i=1 TO 4
2220 GOSUB 2070
2230 PEN 2:FOR n=10 TO 30 STEP 2:LOC
ATE n,10:PRINT q3$:LOCATE n,11:PRINT
 q4$:LOCATE n,15:PRINT q3$:LOCATE n,
16:PRINT q4$:FOR rt=1 TO 30:NEXT rt:
LOCATE n,10:PRINT"  ":LOCATE n,11:PR
INT"  ":LOCATE n,15:PRINT"  ":LOCATE
 n,16:PRINT"  ":NEXT n
2240 NEXT i
2250 PEN 1:LOCATE 13,18:PRINT"PULSA
LA TECLA A":LOCATE 11,20:PRINT"PARA
JUGAR OTRA VEZ"
2260 IF INKEY(69)=0 THEN RUN ELSE GO
TO 2260
```







TE OFRECE LO MEJOR DE LOS MEJORES

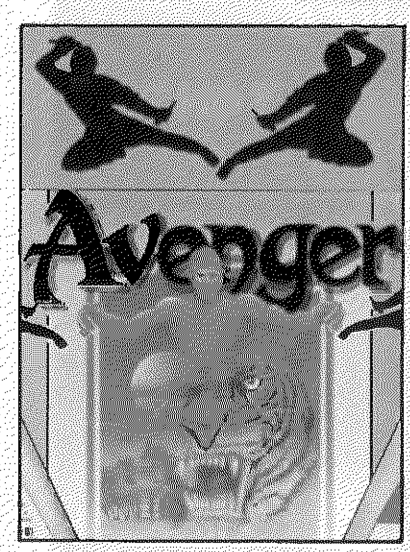
Para que lo tengas fácil. Aquí tienes una selección de los mejores juegos que puedes encontrar. Cualquiera de ellos tiene la garantía de calidad y adicción que esperas encontrar en un videojuego. Para no llevarte "sorpresas"... elige un juego distribuido por ERBE en cualquiera de las buenas tiendas de informática.





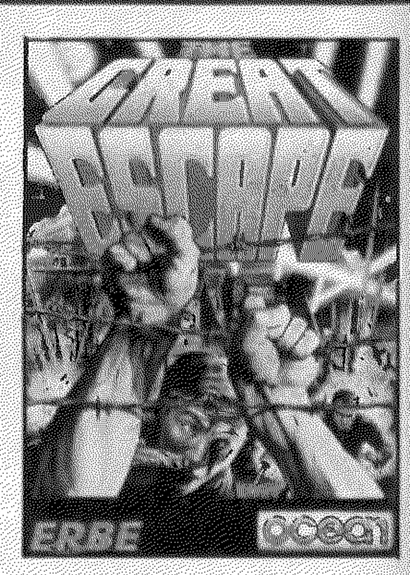
TOP-GUN (Idolos del Aire)

Top Gun te coloca en la cabina de un TOMCAT F-14. Gráficos vectoriales y la pantalla dividida permiten a 1 ó 2 jugadores luchar entre ellos o contra el ordenador. Tus armas en este combate, no apto para cardíacos, son misiles guiados por el calor, y una ametralladora de 20 mm.



AVENGER (Way of The Tiger II)

Primero fue "Way of The Tiger" en donde tuviste que demostrar tus habilidades para convertirte en Ninja. Ahora es "Avenger", en donde tendrás que demostrar que, además de fuerte, eres hábil e inteligente para conseguir vencer al Gran Guardián. Buena suerte... sólo los bravos sobrevivirán.



GREAT ESCAPE (Gran Escapada)

Alemania 1942. La guerra ha estallado y tú has sido capturado y condenado en un campo de concentración. Tu deber es escapar, pero no te resultará fácil. Necesitarás planear cuidadosamente las acciones utilizando toda tu astucia para escapar con vida del campamento.

BATMAN

AMSTRAD PCW

Desde la azotea del rascacielos, la estática figura observa el tranquilo transcurrir de la noche. Pero aunque intenta disfrutar de ello, un torbellino de ideas bombardea su mente. Antes pensaba que su misión sería más simple, sin embargo, ahora que conocía las tretas de su enemigo, sabía que tendría que pelear fuerte y utilizar todos sus superpoderes para recuperar su máquina más impresionante, el Batimóvil. Sí, Batman lucharía hasta el fin para recuperar esta maravilla de las manos del odioso Arlequín.

Con un perfecto y rápido vuelo se dirigió hacia los suburbios de la ciudad, donde sabía que encontraría al cínico **Arlequín** esperándole para tomar la revancha de la derrota que le había inflingido.

Pero también sabía que no le esperaría en campo abierto. Sin duda, el Arlequín se escondería en su cueva repleta de mortíferos ingenios y trampas para jugar con todas las cartas a su favor.

Cuando llegó a la oscura entrada, no vaciló al entrar, puesto que confiaba plenamente en sus sentidos murciélagos, y aunque éstos eran formidables, no pudo darse cuenta de lo rápido que el suelo se abría bajo sus pies.

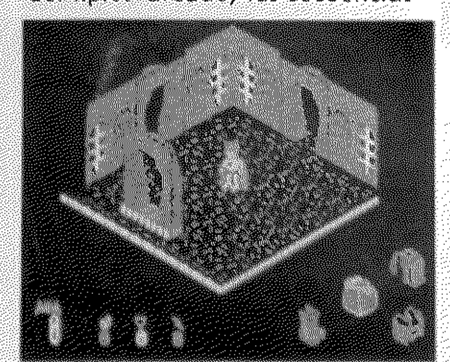
Consiguió aterrizar bien, gracias a su agilidad, mientras pensaba que era ya el momento de la acción.

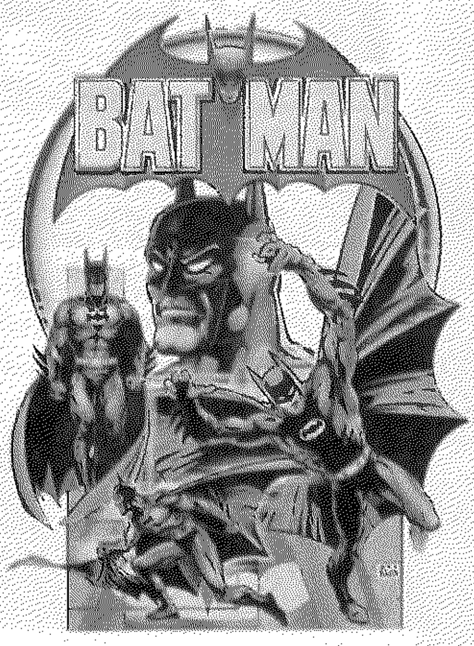
Y si en ese instante **BATMAN** empezaba su aventura, nosotros en éste empezamos a relatar uno de los juegos que podemos tildar de excelente.

Las peripecias de este hombre-murciélago se desarrollan en un tipo de escenario gráfico que en su tiempo, y aun hoy en día, causó furor; estamos hablando de la técnica de **«filmation»**, que inició sus pasos con el conocido KNIGHT LORE.

Algunos de los juegos que siguieron después se conformaron con copiar esta magnífica técnica, otros han intentado mejorarlos. **BATMAN** se encuentra entre estos últimos.

Lo primero que tenemos que destacar de este juego, bueno, más bien de sus creadores, es el alarde imaginativo del que han hecho gala. Si bien, el tema es el del típico arcade, las secuencias gráficas en que éste se desarrolla son de una originalidad y simpatía importante.





Todo esto se ha podido elaborar, y muy bien por cierto, debido a la personalidad del enemigo de **BATMAN**; el malévolo **Arlequín**.

En su cueva podremos ver los seres más estrambóticos; extraños perros, cabezas andantes, diablos que parecen de Tasmania, etc.

También son bonitas y originales las salas por donde veremos discurrir todos estos entes, ya que la cueva del **Arlequín** es una singular mezcla entre parque de atracciones y tétrico castillo medieval.

BATMAN tiene que ir con los ojos muy abiertos para evitar los peligros que se ven, pero no puede olvidar que su oponente es amigo de utilizar escalones que desaparezcan al pisarlos, trampas en los pasadizos, gases letales, y mucho más...

De la parte positiva, que no debemos olvidar, tenemos que nuestro héroe podrá utilizar cuatro de sus especialísimos aparatos, si consigue recogerlos también de las salas donde **Arlequín** los ha dejado, claro está.

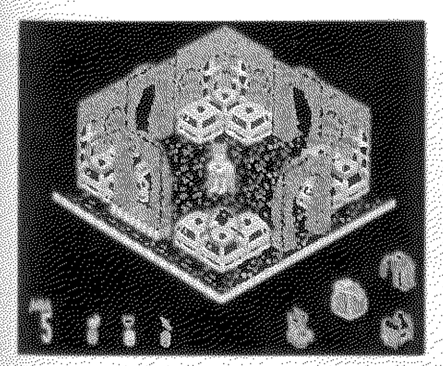
Las poderosas botas le permitirán dar saltos de gran magnitud; con la cesta podrá recoger determinados objetos imprescindibles para su misión; con su propulsor iónico aumentará la velocidad de salto y podrá planear; así como su cinturón antigravedad hará más lentas las caídas.

Cuando consigamos que nuestro superhéroe reúna este portentoso equipo, sus dificultades se reducirán notablemente, aunque todavía le quedará una ardua tarea debido a que el maquiavélico **Arlequín** ha dividido en siete partes de su coche, escondiendo cada una de ellas en una habitación de su inmensa cueva.

Así que **BATMAN** comenzó a caminar por las extrañas habitaciones, preguntándose cuánto tardaría en encontrar el primero de estos útiles aparatos. Afortunadamente, al cabo de poco rato vio en una sala la cesta, custodiada por un ente diabólico, lo esquivó hábilmente y cogió su bolsa. Un par de habitaciones después consiguió encontrar el lugar donde estaban las botas; entró y, después de hacer una demostración de equilibrio para no caer al vacío, cogió las mismas.

Aumentando su velocidad, pero sin dejar de memorizar el recorrido, **BATMAN** siguió recorriendo el galimatías que era la cueva de **Arlequín**.

Y así fue recorriendo otra porción de la cueva hasta que llegó a la habitación donde estaba el propulsor y se lo colocó con una sonrisa. Ya sólo le



quedaba encontrar uno de los portentosos aparatos de su equipo.

Atravesó un extraño puente que se deshacía al pisarlo. Siguió reconociendo habitaciones, y al abrir la puerta de una de ellas vio lo que tanto había buscado: el cinturón.

Pero por la prisa que empleó para ir a recogerlo, no se percató de que, una vez más, el suelo desaparecía bajo sus pies. Pero en esta ocasión el propulsor consiguió que pudiera aterrizar sin consecuencias.

En el suelo de la habitación se elevaba una plataforma que parecía llevar a pisos superiores, **BATMAN** saltó a ella, y mientras subía, descubrió otra plataforma donde estaba la última, y por lo tanto anhelada, pieza de su equipo. **BATMAN** sabía que con éste completo su misión ya casi estaba realizada.

Otro factor que hará más fácil o difícil nuestra tarea serán las figuritas, con forma de **BATMAN**, que Arlequín ha colocado en algunas estancias y corredores.

Unas consiguen que nos movamos con más rapidez; otras, que podamos hacer 10 super saltos; las terceras, que nuestros enemigos se vuelvan inofensivos; así como las cuartas y quintas dan y quitan una vida respectivamente.

Después de tomarse un respiro, **BATMAN** empezó a realizar la búsqueda del objeto que realmente le había llevado allí, el fantástico Batimóvil.

La primera pieza del mismo, una aleta, se hallaba en una de las habitaciones inferiores. Para cogerla desdeñó una falsa escalera de cubos, que era una trampa, y colocó una pieza que encontró bajo el cubo que había junto a la aleta y saltó. El cubo se deshizo y la aleta cayó al suelo, la cogió y siguió su camino.

La segunda la encontró tras una intrincada ascensión en una solitaria habitación. Si hubiera caído, tendría que haber vuelto a subir, perdiendo con ello un tiempo precioso.

Pero esto no ocurrió, y raudo fue a buscar la tercera pieza. Esta apareció en una excéntrica sala donde subían y bajaban dos émbolos. Como era muy accesible el lugar donde estaba, no le supuso mucho esfuerzo el cogerla.

La quinta, otra aleta, se encontraba en una habitación, donde todo se había planeado como una trampa mortal. Nada más entrar en ella **BATMAN** oyó un fuerte zumbido. Levantó los ojos, y apenas tuvo tiempo de apartarse para esquivar la enorme piedra que le venía encima. Después de recuperarse de la impresión, se dispuso a abandonar la habitación, que parecía no contener nada, cuando un brillo azulado en un laberinto de zetas llamó su atención. Evitando tocar las paredes llegó hasta el lugar donde se veía el fulgor y descubrió que se trataba de la quinta pieza.



No muy lejos de la anterior encontró la sexta. Para llegar a la sala donde estaba había que recorrer un corredor plagado de obstáculos. Cualquier error conducía al abismo. **BATMAN** logró pasar sin percance alguno, gracias a su excepcional sentido del equilibrio, y recogió la pieza que se encontraba en lo que parecía ser la imitación de un castillo.

Con la certeza de poder conseguir recuperar su Batimóvil, **BATMAN** prosiguió la búsqueda de la séptima pieza.

Pero ésta se encontraba muy bien escondida y requirió que nuestro héroe tuviera que recorrer innumerables habitaciones. Empezaba el desánimo a hacer mella en él, cuando entró en una habitación con tres teteras y una serie de cubos formando puente sobre los que descansaba una esfera con una interrogación. **BATMAN** comprendió de repente y se puso a trabajar.

Situó dos teteras junto a la puerta formando tope. Trepó por el puente de cubos y empujó la bola que rodó hasta detenerse junto a las teteras. Con sumo cuidado separó éstas y se situó tras la bola. Tomó impulso y la lanzó a rodar bajo el puente de cubos. La bola chocó contra un objeto y lo arrastró al exterior. La última pieza del coche estaba en su poder.

Como se puede ver, en este juego no falta emoción y actividad, ésta incluso la podríamos definir como febril.

Si hablamos a nivel gráfico, no podemos olvidar que en esta ocasión el juego va dirigido a los PCW, y que en éstos el monitor es de fósforo verde, o sea, debemos olvidarnos del color, o por lo menos de todo lo que no sea verde.

Si bien esto no podría importar en otro juego, no ocurre en éste, debido a que en los CPC se desarrolla en pantallas casi monocromas, así que no nos perdemos un gran colorido.

El movimiento sí que es realmente bueno, rápido y divertido.

Es realmente gracioso observar cómo se pone nuestro héroe en **«jarras»** y golpea el suelo nerviosamente con su pie, cuando le hemos hecho pararse durante un rato. Es como si dijera: **«¿a qué esperas?»**.

Si a este acertado movimiento le añadimos unos dibujos muy buenos y una trama, aunque ya clásica, llena de peripecias, formaremos un **«cocktail»** que logrará entretenernos.

Los elementos principales del joystick son: la base o carcasa, la empuñadura, el sistema de articulación y los elementos eléctricos. Cada uno de ellos juega un importante papel y solamente un buen resultado conjunto podrá ofrecer la garantía de transferir eficaz y rápidamente al ordenador vuestos movimientos.

La empuñadura (stick), es el primer eslabón de la comunicación, las características particulares de este elemento son las que confieren al periférico una buena parte de su comodidad de uso, por ello, su diseño se realiza en base a conceptos anatómicos, si bien sólo son unos pocos los modelos del mercado que logran una buena adaptabilidad a la mano del jugador y así permitir un uso prolongado consiguien-

De ordinario la elección de periféricos para ordenadores suele realizarse tras la valoración personal de las características técnicas del producto; sin embargo, en los joysticks la mayoría de éstos son subjetivos y sólo la experiencia de uso nos puede conducir a una acertada valoración. Con este artículo intentamos ayudaros informándoos de los que hay en el mercado.

JOYSTICK



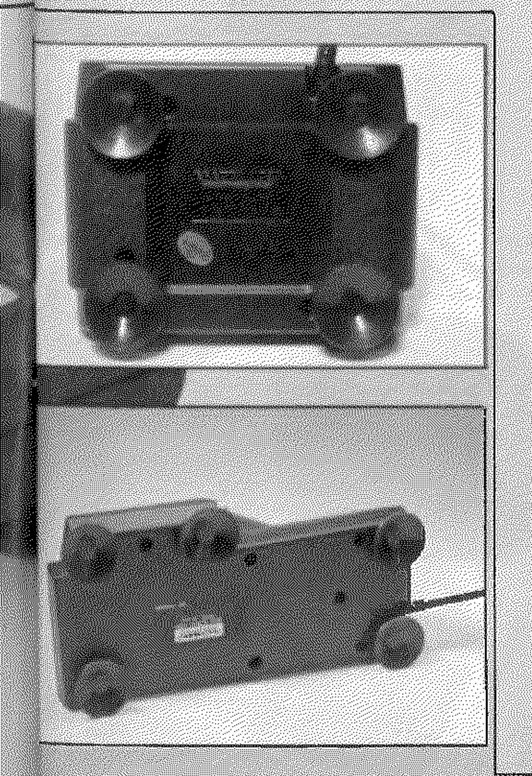
do con ello la mínima fatiga muscular.

La operatividad de la empuñadura es poder efectuar los desplazamientos correspondientes a las cuatro posiciones (N, S, E y O) que posteriormente analizará el ordenador y sus combinaciones de movimientos diagonales (NO, NE, SO y SE).

Una ròtula esférica sirve de elemento de unión entre el stick y la carcasa, unión que debe permitir un suave desplazamiento del stick, esta suavidad de desplazamiento y la ausencia de holguras en el mecanismo determinará la precisión del joystick. El otro extremo de la empuñadura queda situado entre cuatro microruptores que son activados o desactivados al alcanzar el stick la situación correspondiente. La construcción de estos contactos varia desde simples laminas flexibles hasta microruptores mecánicos (reconocibles por su «clic» característico) e incluso simples circuitos impresos superpuestos que generalmente están integrados en una pieza de plástico que a veces llega a efectuar las funciones de la rótula.

A fin de que la empuñadura retorne a su posición central (neutra), un sistema de resortes o masas elásticas (gomas) unidas a la parte inferior de la rótula realizan tal operación.

La robustez de todo este conjunto mecánico es un factor determinante para aquellos que día a día se enfrentan a duras batallas.



MECANISMOS DE DISPARO

Para la ejecución de disparos (o saltos) hay gran variedad de versiones, para ello se dota al joystick de varias posibilidades, tantas como hábitos puedan tener los jugadores.

Generalmente éstos se producen al pulsar algún botón de diferentes formas y tamaños que se encuentran distribuidos en la empuñadura y/o la base.

Lo más habitual es que en la empuñadura haya al menos un pulsador al alcance del dedo pulgar y a veces se complemente con otro a modo de gatillo, accionable con el indice. También, la carcasa puede tener uno o varios pulsadores de efecto semejante a los del stick, todos estos pulsadores están concentrados eléctricamente en paralelo pudiendo efectuarse el disparo desde cualquiera de ellos. Algunos modelos están complementados con un interruptor de disparo permanente que sirve de gran ayuda en los juegos de trepidante acción (salvo en aquellos en los que la energía es en función inversa a los disparos).

En este capítulo es destacable la importancia de la recuperación de todos y cada uno de los ruptores a fin de que ésta no ralentice la sucesión de disparos en ráfaga.

LA SUJECION

La carcasa o base del joystick cumple una doble misión, una, la de albergar en su interior todos los mecanismos descritos anteriormente, la otra, la de ofrecer una gran sujeción del conjunto a la superficie de la mesa en el caso de estar prevista para ello o la de acomodarse a la mano en aquellos tipos de joystick diseñados para este modo de utilización.

En el primer caso la sujección del joystick viene realizándose a base de unas ventosas que, dispuestas en su parte inferior, los inmovilizan suficientemente. La mayoria de los modelos analizados poseen cuatro ventosas con la única excepeción del Quick Shot V que utiliza cinco, por otra parte parecen haberse puesto de acuerdo todos los fabricantes en el tamaño de éstas..., todas ellas son de 30 mm de diámetro. Sin embargo, el grado de sujeción no sólo está en función del número de ventosas sino que también tiene su importancia la base de sustentación que éstas proporcionan y la longitud del stick, puesto que a mayor longitud de éste mayor empuje habrá de soportar la base.

CONEXION

La descripción del joystick queda completada con una pequeña alusión al cable de conexión en el que cabe destacar la importancia de una longitud que permita su manipulación a una distancia apropiada que casi todos poseen.

Por otro lado, la calidad del cable de conexión aunque no es influyente en la manipulación del periférico sí puede ser indicativo de la calidad general.

DESARROLLO

En los últimos años la evolución de este periférico ha alcanzado cotas muy altas pero sigue siendo el diseño clásico el de mayor difusión, desarrollándose multitud de nuevos modelos más ergonómicos, cómodos y duraderos, incorporando a ellos ingeniosos complementos al efecto como bases más amplias (Quick Shot III y V, Cobra), mini teclados para introducción de niveles de dificultad y número de jugadores (QSV), utilización de mecanismos de alta calidad (baza que gana el Cobra), etc.

En cuanto a los modelos más avanzados éstos presentan innovaciones realmente ingeniosas y de conceptos absolutamente distintos de lo habitual, bien que su aplicación debe ser enjuiciada según cada particular.

Modelos como el Cheetach de mando a distancia evitan el a veces engorroso cable de conexión, si bien puede «jugarosla» durante una partida si en un momento de exaltación lo desviamos de la dirección del interface receptor. Sistema muy similar utiliza el Quick Shot VII, pero sin dejar a un lado el cable conector.

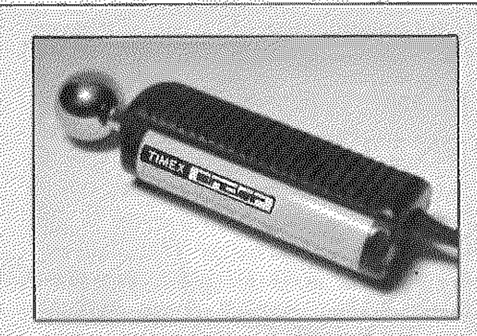
El Joycard, es un reducido teclado que incorpora un joystick y un par de pulsadores en simulación a las «maquinitas» de los bares.

Quizás el modelo de más impacto

visual sea el último de la extensa saga de Quick Shot, la versión nueve, una enorme bola de 10 cm de diámetro movible en cualquier dirección, dotada de una gran precisión y que incorpora dos teclas de gran dimensión para disparo y complementado con un par de interruptores que permiten las opciones de fuego automático e inversión de sentido de desplazamiento, haciendo posible distintas situaciones del aparato.

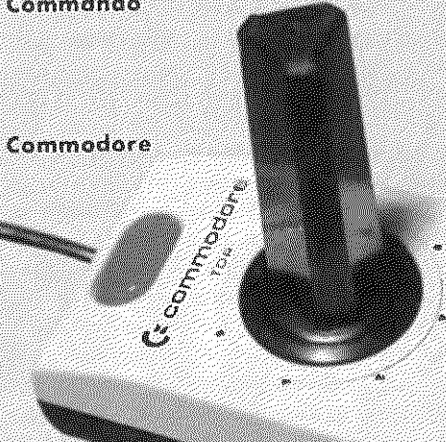
EL PRECIO DEL PODER

Realizar un análisis de precios correspondientes a cada modelo de joystyck de una forma fehaciente no es tarea fácil dado que generalmente se encuentran formando parte de atractivas ofertas, cuando no se incluyen en la compra del ordenador, pero orientativamente oscilan alrededor de las 2.000/3.000 ptas. los modelos más convencionales, alcanzando 10.000 y 12.000 los modelos más precisos y/o sotisficados.

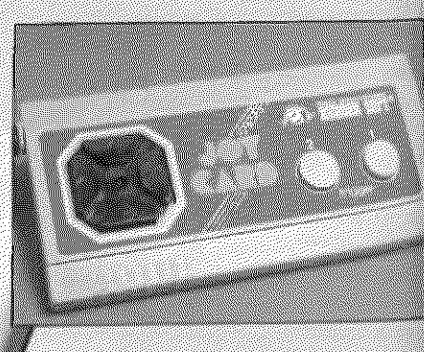


Commando

Joystick



Commodore



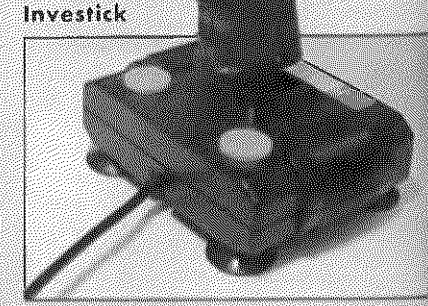
Joycard



Capitán Grant



Gun Shot



Investick



Cheetah

Cobra



Quick Shot IX




Quick Shot III
Quick Shot I
Kempston
Quick Shot
VII
Kempston
3000
Quick Shot II
Quick Shot V
Konix
Superstick
Proto
Toshiba

CARACTERISTICAS TECNICAS

MODELO	EMPUÑADURA Tipo	Disparo	Altura	BASE Sujeción	Disparo	DIMENSIONES (cm) Totales	Cable
CAPITAN GRANT	Anatom.	1 pulsador	12	Manual	1 pulsa.	13×10×16	125
CHEETAH	Anatom.	1 pulsador		Manual		16×6.5×2.5	Mando distancia
COBRA	Anatom.	2 pulsad. 1 gatillo	16	4 ventosas		14×12.5×24	135
COMMANDO	Anatom.	1 pulsador	3			4.5×3×17	170
COMMODORE	Lisa	1 pulsador	7,5	Manual	1 tecla	10×7×11	127
GRAN CAP. II	Anatom.	1 pulsador 1 gatillo	12	Manual	2 pulsad. +AUTO	12×12×16.5	125
GUN SHOT	Anatom.	1 pulsador	13,5	4 ventosas	1 pulsad.	13×11×16	125
INVESTICK	Anatom.	1 pulsador 1 gatillo	13,5	4 ventosas	2 pulsad.	13×10×18	104
JOYCARD	de bola		4,5	sobremesa	2 pulsad.	18×10×6.5	27
JOYSTICK	Anatom.	1 pulsador 1 gatillo	13	4 ventosas	2 pulsad.	13.5×10×17	124
KEMPSTON	de bola		7	Manual	2 pulsad.	11×5×9×11	120
KEMPSTON 3000	Anatom.	1 pulsador 1 gatillo	12	Manual	1 tecla	13×7.5×16	160
KONIX	de bola	1 gatillo	6			13×8×10	135
PROTO	Anatom.	1 pulsador 1 gatillo	12	4 ventosas	1 pulsad.	12×12×15.5	126
QUICK SHOT I	Anatom.	1 pulsador	12	4 ventosas	1 pulsad.	11×9×16	126
QUICK SHOT II	Anatom.	1 pulsador 1 gatillo	13.5	4 ventosas	1 pulsad. +AUTO	13×9.5×17.5	126
QUICK SHOT III	Anatom.	1 pulsador 1 gatillo	12	5 ventosas	2 pulsad. TECLADO	19×10×17	126
QUICK SHOT V	Anatom.	1 pulsador 1 gatillo	12	4 ventosas	1 tecla	19×9.5×17	126
QUICK SHOT VII	Disco	2 gatillos				12×9×2.5	126
QUICK SHOT IX	Esfera 10 cm		10	4 ventosas	2 teclas +AUTO	22×14.5×12	121
SUPERSTICK	Cilindro	1 pulsador	10	Manual		9×9×13	157
TOSHIBA	Anatom.	1 pulsador	11	Manual	1 tecla	13×10×18	104

VALORACION

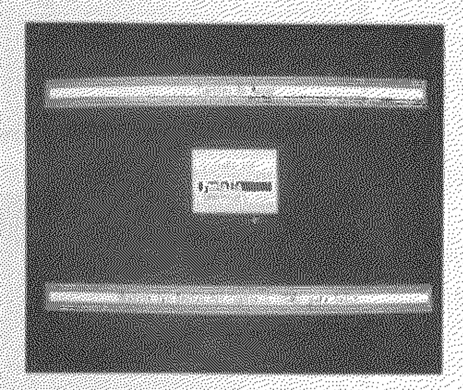
MODELO	GRADO DE SUJECION	SUAVIDAD DE MOVIM.	SUAVIDAD DE DISP.	ADAPTACION A LA MANO	ROBUSTEZ MECANIS.	COMODIDAD USO PROLONG.
CAPITAN GRANT		***	***	****	***	**
CHEETAH		**	***	****	*****	***
COBRA	*****	****	*****	*****	*****	*****
COMMANDO		***	***	****	****	***
COMMODORE		*	**	*	***	*
GRAN CAP. II	***	****	***	****	***	****
GUN SHOT	***	****	****	****	***	****
INVESTICK	***	***	***	**	**	**
JOYCARD		****	****	****	****	****
JOYSTICK	***	*****	***	***	***	***
KEMPSTON		****	****	****	****	**
KEMPSTON 3000		**	**	**	***	**
KONIX		*****	****	*****	****	***
PROTO	***	***	***	***	****	****
QUICK SHOT I	***	****	****	****	***	****
QUICK SHOT II	***	****	****	****	***	****
QUICK SHOT III	*****	****	****	****	****	*****
QUICK SHOT V	****	****	****	****	****	*****
QUICK SHOT VII		***	****	****	****	****
QUICK SHOT IX	*****	****	*****	*****	*****	*****
SUPERSTICK		***	**	**	**	****
TOSHIBA		*****	*****	*****	***	***

Contabilidad General, Vencimientos

Fco. Javier Barceló T.

Dentro de los programas de gestión, los programas de contabilidad son los que más abundan. Y el Amstrad PCW no va a ser una excepción. Por eso, para facilitar la elección entre los diversos programas, hoy, AMSTRAD Semanal analiza Contabilidad General + Vencimientos.

Para dar una idea clara de este programa, seguiremos la misma norma empleada en otros artículos, que no es más que describir exhaustivamente todas sus posibilidades. Cuando se busca un programa de contabilidad, normalmente se tienen las cosas muy claras, sobre qué se necesita y qué no se necesita, por lo que lo mejor para elegir bien es saber qué es lo que hace y cómo lo hace cada programa. Esperamos así facilitar su elección.



Para empezar, el programa no ofrece problemas de instalación. Se teclea el habitual comando SUBMIT, seguido del nombre del programa *(contable)*, y aparece la primera pantalla, que advierte sobre la necesidad de copiar al disco de datos los ficheros PLANCO.DAT y CONTAB.AUX para que el programa no se cuelgue, o dé errores. El programa de instalación inicial copia todos los programas en el disco virtual M: permitiendo así introducir el disco de datos en la unidad A: y funcionar así toda la sesión.

Una vez instalado, aparece el menú principal, donde están todas las opciones del programa. Naturalmente, lo primero que hay que hacer es elegir la opción INSTALACION DEL PROGRAMA, para adecuarlo a nuestras necesidades. En esta opción se elige el tipo de papel que se va a usar, el nombre que aparecerá en la cabecera de los listados, el número de UNIDADES de disco de las que disponemos y luego pregunta el número de subcuentas que vamos a necesitar. Este dato es vital, porque según el número de cuentas que introduzcamos, podremos disponer de más o menos asientos. Una vez introducida la cifra, nos da el número máximo de apuntes que admite.

Una vez hecho esto, el programa vuelve al menú principal. Las demás opciones de que dispone son:

- **Asientos contables.**
- **Plan Contable.**
- **Listado de Cuentas de Mayor.**
- **Balance de Sumas y Saldos.**
- **Cuenta de Resultados.**
- **Saldo del IVA.**
- **Vencimientos.**

Siguiendo un orden cronológico, lógicamente lo primero será dar de alta las cuentas, mediante la opción PLAN CONTABLE. Esta opción dispone de un submenú con las siguentes opciones:

- **Introducción de Cuentas.**
- **Modificación de Cuentas.**
- **Baja de Cuentas.**
- **Consulta de Cuentas.**
- **Listado del Plan Contable.**
- **Ordenación de Cuentas.**
- **Actualización de Saldos.**
- **Poner los Saldos de las Cuentas a cero.**

Al seleccionar la introducción de cuentas, pasamos a la pantalla que nos pregunta el número y el nombre de la cuenta. Si introducimos un número de cuentas ya existente, nos avisa de ello. Existen cuatro niveles de cuentas, con uno, dos, tres y siete dígitos respectivamente, y, además, el programa viene ya con parte del plan contable introducido, cosa que es de agradecer. Naturalmente, para los que utilicen planes especiales, se pueden modificar todas las cuentas, vengan con el programa o las hayamos introducido nosotros. A la derecha de la pantalla, viene el espacio disponible para seguir introduciendo cuentas, por lo que es muy importante calcular bien antes de introducirlo en la instalación del programa.

Como las cuentas no tienen por qué haberse metido en orden, después de dar de alta el plan conviene utilizar la opción ORDENACION DE CUENTAS, que las clasifica numéricamente. Para prevenir cualquier fallo, es conveniente realizar antes una copia de seguridad de los ficheros.

Las demás opciones permiten modificar una cuenta determinada y dar de baja una cuenta, siempre que ésta tenga saldo cero, dado que si tiene movimientos no se puede anular. Después de bajar alguna cuenta, también es necesario ejectuar la opción de ordenación de cuentas, para que el programa reconstruya el índice de las mismas.

La opción CONSULTA DE CUENTAS permite consultar una cuenta o grupo de cuentas, con el total de sus movimientos y su saldo. Al preguntar al programa el número de cuenta, si se le da un número de una, dos o tres cifras, presenta todas las cuentas que empiecen por el número dado, mientras que si se le da un número de siete cifras, presenta los movimientos de esa cuenta solamente.

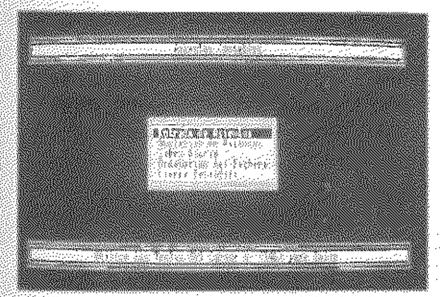
La opción LISTADO DEL PLAN CONTABLE proporciona un listado de todas las cuentas del plan, ordenadas numéricamente.

La opción ACTUALIZACION DE CUENTAS pasa los saldos de las subcuentas a los de las cuentas de nivel superior, dejando los saldos de las subcuentas a cero. Antes de realizar este proceso es imprescindible hacer copias de seguridad de los ficheros, así como listar las cuentas del Libro Mayor y el Balance de Sumas y Saldos a nivel 7.

Volviendo al menú principal, una vez creado el plan contable, se puede empezar a introducir asientos. Al escoger la opción ASIENTOS CONTABLES, aparece un submenú con las siguientes opciones:

- **Entrada de Asientos.**
- **Anulación de Asientos.**
- **Libro Diario.**
- **Ordenación del Fichero.**
- **Cierre Periódico.**

La introducción de Asientos, eligiendo dicha opción, resulta muy sencilla. Primero asigna



automáticamente el número de partida, dato este que no se puede modificar. Luego pregunta la fecha, comprobando si es correcta, el Número de Subcuenta, si ésta existe, visualiza su nombre en la última línea, y pregunta el concepto, y si la partida es de Debe o Haber. El programa comprueba que la cuenta introducida exista, así como la fecha, aunque los apuntes no tienen por qué ser introducidos cronológicamente, puesto que la opción ORDENAR FICHERO los reordena de esta manera. Si se pulsa RETURN en un campo vacío, aparece el valor del apunte que está inmediatamente encima del actual, evitando así tener que repetir —por ejemplo— el concepto varias veces. En el importe, si éste fuera negativo, admite el signo menos y, además, puntúa a la inglesa. Es decir, las comas de separación de miles son puntos, y el punto decimal es una coma. La parte inferior de la pantalla mantiene informado sobre el saldo actual del diario, y si se desea salir de esta opción estando la contabilidad descuadrada, un mensaje lo avisa, pero permite salir volviendo a pulsar SAL.

La pantalla tiene una última columna, llamada Ok. donde si el apunte es conforme, al pulsar RETURN se valida. En caso contrario, pulsando SAL se retorna al principio. Pero además de esto, poniendo en esta columna una V, marca el asiento como el vencimiento pendiente, pregunta la fecha de vencimiento del apunte. Esto posibilita el control de los vencimientos de manera extremadamente cómoda.

El programa no tiene prevista la opción de modificar asientos, por lo cual dado el caso, hay que anular el apunte incorrecto, e introducir de nuevo el apunte correcto. Dado que el programa ordena los apuntes por fechas, eso no resulta un gran problema, puesto que el diario permanece igual.

La opción de Listar el Libro Diario permite elegir entre qué dos fechas se desea, y si se desea por pantalla o por impresora. Si se elige el listado por impresora, pregunta cuál debe ser el número de la primera página, así como la fecha que se desea que aparezca en el listado.

Ya se ha dicho que la opción ORDENACION DE ASIENTOS ordena los apuntes cronológicamente, no obstante, si se ha efectuado un cierre periódico en una fecha, y se introduce un asiento ANTERIOR a dicha fecha, el saldo de las cuentas no resultará exacto.

La opción CIERRE PERIODICO posibilita la introducción de más apuntes, cuando se ha alcanzado el límite de la capacidad del disco. Para ello, actualiza los saldos de las cuentas, borra los asientos y conserva los saldos. Es de vital importancia que se haya hecho una copia de seguridad del fichero de asientos, puesto que de lo contrario, sería imposible remediar una equivocación anterior a dicho cierre. Resulta conveniente, según el número de apuntes que se realicen mensualmente, hacer coincidir éste con algún proceso de actualización de saldos, para dar mayor coherencia a la estructura de los ficheros. Esta es la última opción de ASIENTOS CONTABLES.

Retomando pues, el menú principal, la siguiente opción es el LISTADO DE CUENTAS DE MAYOR. Esta opción presenta, bien por pantalla o bien por impresora, los movimientos de la cuenta o cuentas que se le soliciten, entre dos fechas dadas. Al igual que en la consulta de asientos, si al pedir la primera y última cuenta se introducen cuentas de uno, dos o tres dígitos, el listado se realiza a partir de estas cuentas y todas las de niveles inferiores. De introducir sólo números de subcuentas, no listará los niveles anteriores. Además, si se contesta en blanco a alguna de las dos preguntas, el programa interpreta que se desea desde la primera cuenta y/o hasta la última. Si el listado es por impresora, también pregunta la fecha que se desea que figure en el listado.

La siguiente opción del menú principal es el BALANCE DE SUMAS Y SALDOS. Este listado, permite elegir el nivel de cuentas al que se desea, y su resultado depende directamente de dicho nivel. Si no se ha actualizado el fichero de asientos, se podrá pedir a nivel 7, que es el máximo de desglose. Entonces, imprimirá la suma del Debe y la del Haber, así como el saldo de todas las subcuentas. Pero si se ha hecho ya el proceso de actualización, dado que las subcuentas estarán todas con saldo cero, no tiene sentido. Pidiéndolo a un nivel superior, los saldos de las cuentas reflejarán los movimientos acumulados.

LA CUENTA DE RESULTADOS refleja el resultado económico del período. Tiene tres opciones:

— Aportando stock final.
— Aportando porcentajes sobre ventas.
— Stock inicial = Stock final.

Con la primera opción, el resultado es el más aproximado, dado que sabiendo el stock final el programa obtiene el margen real de beneficio que se está cargando a los artículos vendidos, obteniendo el resultado exacto.

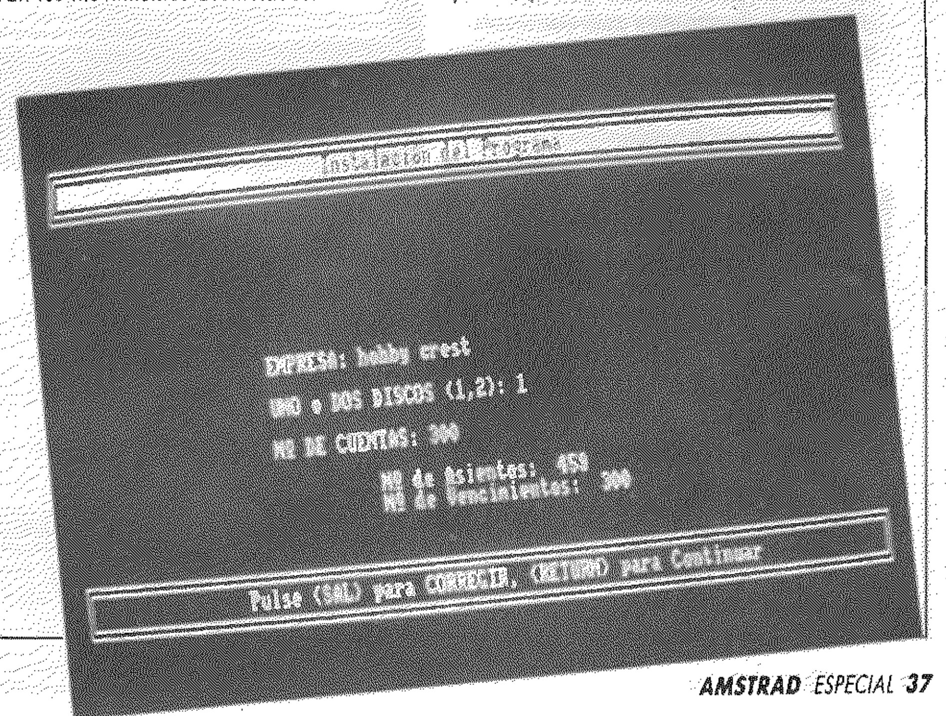
La segunda opción realiza la operación al revés que la primera. Esto es, se le da el porcentaje sobre los ventas y en base a éste se obtiene de manera aproximada el stock final de mercancías.

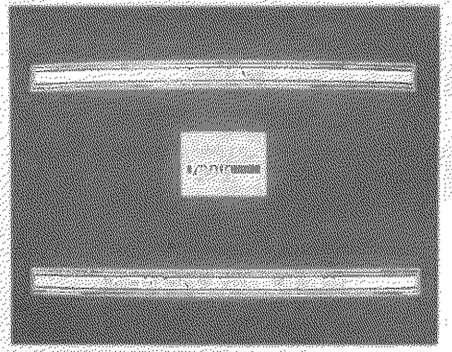
La tercera opción es para cuando no haya stock de mercancías, o sea imposible conocerlo, dependiendo el resultado de esto último. Si hubiera existencias, el resultado no es demasiado fiable, mientras que si realmente no las hay, entonces éste se aproxima más a la realidad.

La siguiente opción es SALDO DEL IVA. Si se ha contabilizado normalmente, realiza un informe de los movimientos de las cuentas que recogen el Impuesto sobre el Valor Añadido, que son:

***473.*— IVA Soportado.**
***474.*— Hacienda deudora por IVA Soportado.**
***478.*— IVA Repercutido.**
***479.*— Hacienda acreedora por IVA Repercutido.**

Para realizar este proceso correctamente, hay primero que actualizar las cuentas, de manera que las cuentas de tres cifras recojan todo el saldo, y el resultado sea correcto. Una vez revisado el saldo de las dos cuentas de IVA, el programa dará el saldo de dicho impuesto, y a cuál de las cuentas de Hacienda





hay que cargarlo. Naturalmente, aparte de los motivos fiscales, es necesario para que el programa funcione correctamente, que todo apunte de IVA pase a través de las subcuentas de estas dos cuentas.

La opción de VENCIMIENTOS, utiliza los apuntes en los que se marcó dicha clave, para ayudar a localizarlos. Dispone de las siguientes opciones:

- **Pendientes de Pago.**
- **Pagados.**
- **Pago de Vencimientos.**
- **Actualizar fichero.**
- **Ordenar Vencimientos.**

La primera opción lista los vencimientos pendientes de pago desde la primera fecha que se le dé, hasta la segunda. Permite elegir si se desea por pantalla o impresora.

La segunda opción, es idéntica a la anterior salvo que lista los pagos ya realizados y vencidos.

La tercera opción sirve para anotar el pago de los vencimientos. Esta opción, presenta una pantalla similar a la de entrada de asientos, y al darle el número del vencimiento, se presenta en la pantalla. Si se está conforme con esto, se le cambia la situación en la columna de la derecha a P pagado. Realizar esto no significa nada a nivel contable. Si se ha pagado, habrá que realizar un asiento en la contabilidad reflejándolo, y además recogerlo en el fichero de vencimientos para que los informes que realice esta opción sean correctos.

Cada vez que se realicen bastantes pagos, y la capacidad del fichero sea pequeña, se utilizará la opción ACTUALIZAR FICHERO, que borrará del mismo todos los apuntes que estén pagados y vencidos simultáneamente, ganando así espacio en disco.

Para no variar, en este menú también hay una opción que permite ordenar cronológicamente los vencimientos, cuyo funcionamiento es igual a los otros que se han visto.

Como conclusiones, la primera sería decir que el programa es muy fácil de manejar, bien presentado y estructurado. Su manejo no ofrecerá complicaciones a nadie. *(A nadie que sepa contabilidad, claro...)*

No obstante, su potencia no es nada del otro mundo. La capacidad de apuntes y cuentas es escasa, y se hacen necesarias las engorrosas maniobras de actualizar, borrar asientos y cambiar discos. En línea, se pueden disponer aproximadamente de 1.600 apuntes, con 500 cuentas y 500 vencimientos simultáneamente, en un disco normal. Naturalmente, si se dispone de la segunda unidad de disco de 760, la capacidad en asientos se dispara a 9.000... y la cosa cambia un poco.

Los listados que efectúa son correctos, faltándole al programa, quizás, un poco más de

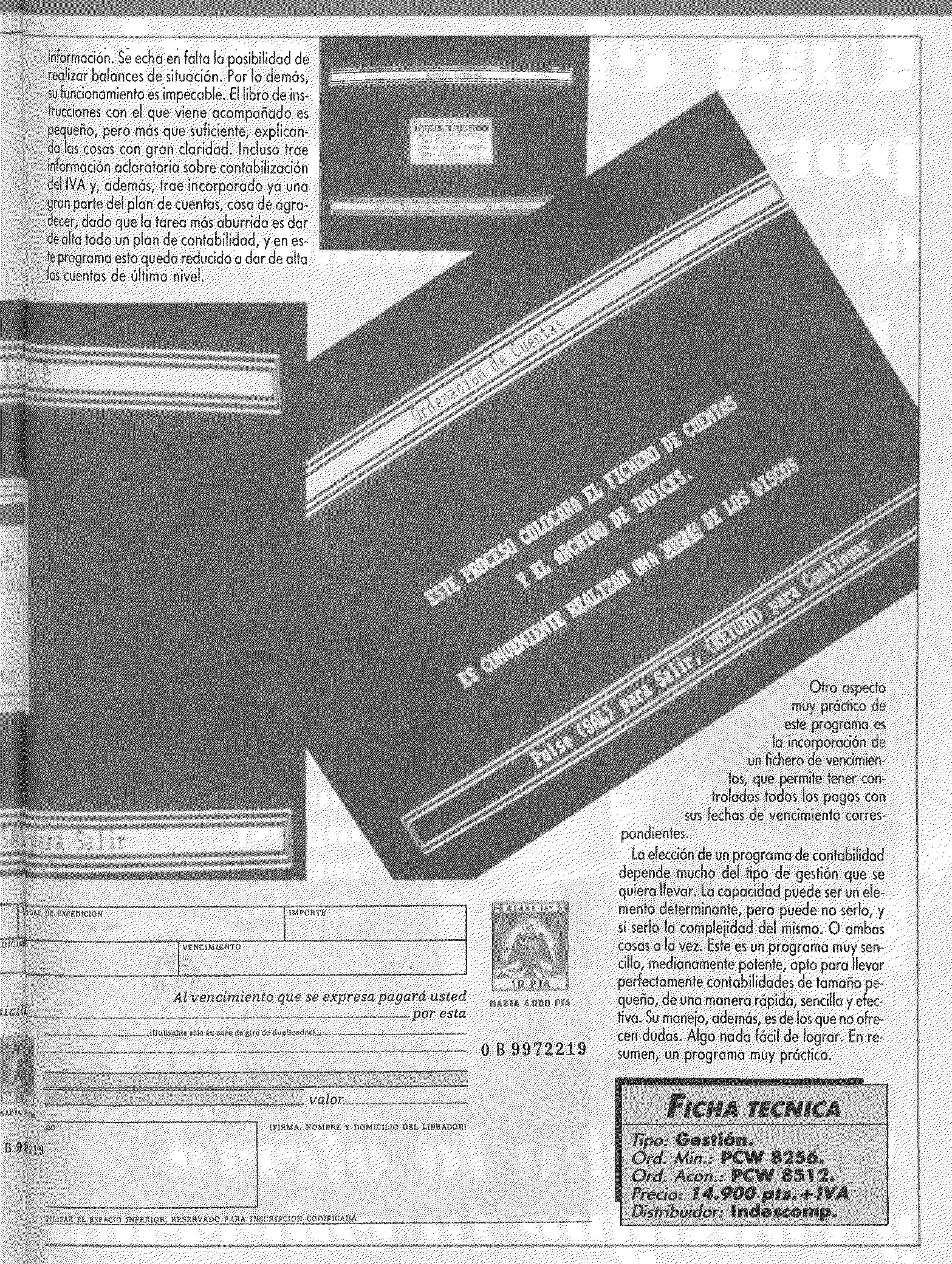
CONTABILIDAD GENERAL

Asientos Contables
Plan Contable
Listado Cuentas de Ma
Balance de Sumas y Sal
Cuenta de Resultados
Saldo de I.V.A.
Vencimientos
Instalación del Prog

Utilice las Teclas del Cursor o



información. Se echa en falta la posibilidad de realizar balances de situación. Por lo demás, su funcionamiento es impecable. El libro de instrucciones con el que viene acompañado es pequeño, pero más que suficiente, explicando las cosas con gran claridad. Incluso trae información aclaratoria sobre contabilización del IVA y, además, trae incorporado ya una gran parte del plan de cuentas, cosa de agradecer, dado que la tarea más aburrida es dar de alta todo un plan de contabilidad, y en este programa esto queda reducido a dar de alta las cuentas de último nivel.



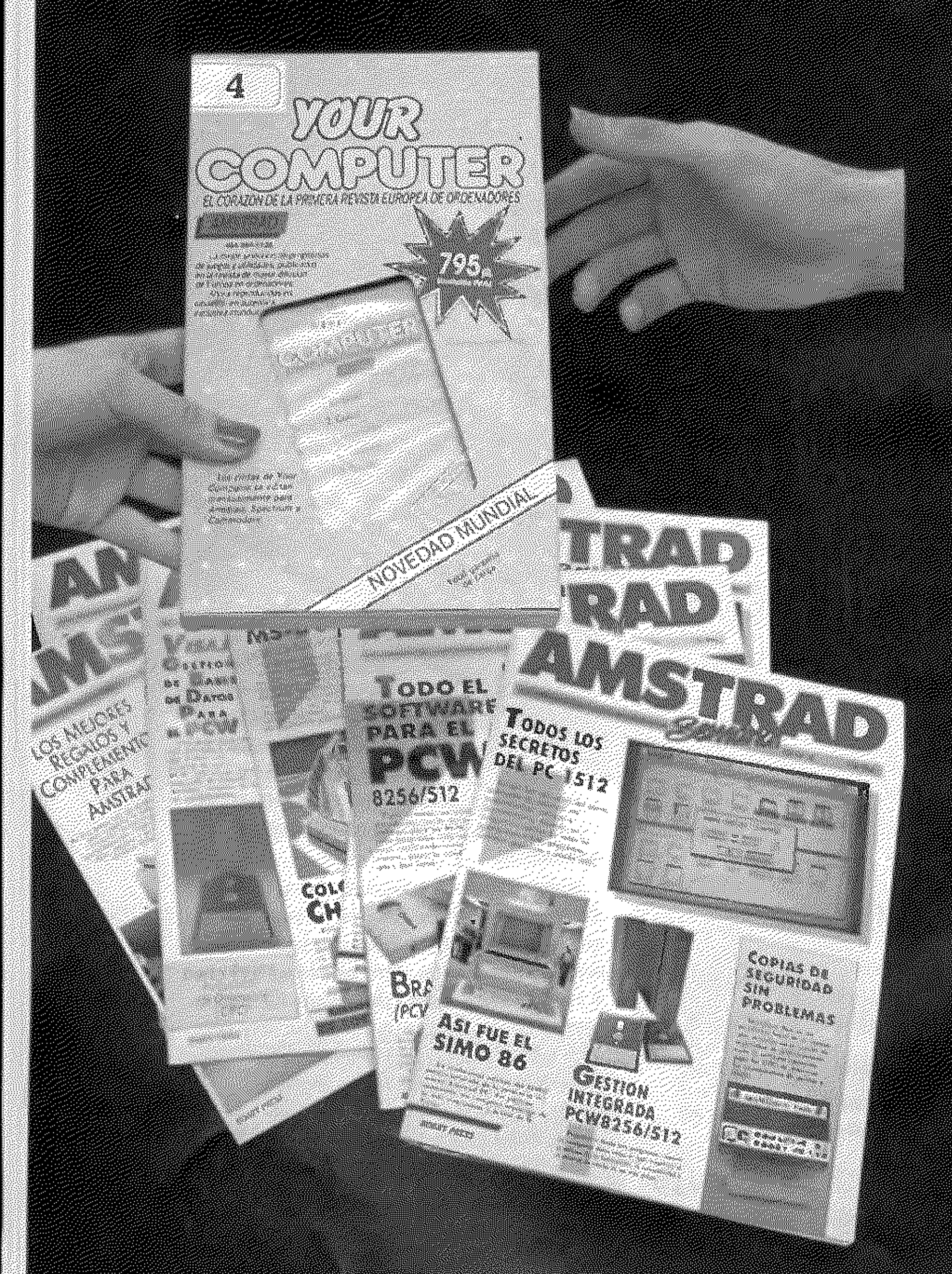
Otro aspecto muy práctico de este programa es la incorporación de un fichero de vencimientos, que permite tener controlados todos los pagos con sus fechas de vencimiento correspondientes.

La elección de un programa de contabilidad depende mucho del tipo de gestión que se quiera llevar. La capacidad puede ser un elemento determinante, pero puede no serlo, y sí serlo la complejidad del mismo. O ambas cosas a la vez. Este es un programa muy sencillo, medianamente potente, apto para llevar perfectamente contabilidades de tamaño pequeño, de una manera rápida, sencilla y efectiva. Su manejo, además, es de los que no ofrecen dudas. Algo nada fácil de lograr. En resumen, un programa muy práctico.

FICHA TECNICA

Tipo: **Gestión.**
Ord. Min.: **PCW 8256.**
Ord. Acon.: **PCW 8512.**
Precio: **14.900 pts. + IVA**
Distribuidor: **Indescomp.**

Una cinta gratis por la compra
de números atrasados



Con la compra de siete números atrasados recibirás gratis una cinta de programas de YOUR COMPUTER (el mejor software inglés), totalmente gratis.

Aprovecha la oferta y consigue tu colección

MICROSOFT-HARD, S. L.

APARTADO 24.399, 08080 BARCELONA. Teléf.: (93) 348 04 07 (9 a 13 y de 16 a 20 horas)

MANTENGA SU AMSTRAD COMO NUEVO CON UNA DE ESTAS PRACTICAS FUNDAS

COD.	ARTICULO	MONITOR	P.V.P.
FC01	FUNDA ORDENADOR CPC 464	(verde)	2.262 ptas.
FC02	FUNDA ORDENADOR CPC 464	(color)	2.262 ptas.
FC31	FUNDA ORDENADOR CPC 472	(verde)	2.262 ptas.
FC32	FUNDA ORDENADOR CPC 472	(color)	2.262 ptas.
FC03	FUNDA ORDENADOR CPC 664	(verde)	2.262 ptas.
FC04	FUNDA ORDENADOR CPC 664	(color)	2.262 ptas.
FC05	FUNDA ORDENADOR CPC 6128	(verde)	2.262 ptas.
FC06	FUNDA ORDENADOR CPC 6128	(color)	2.262 ptas.
FC33	FUNDA UNIDAD DE DISCOS		412 ptas.
FC17	FUNDA ORDENADOR PCW 8256		3.250 ptas.



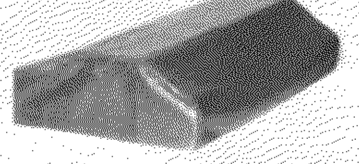
AMSTRAD PCW-8256/8512

COD.	ARTICULO	P.V.P.
FC12	FUNDA IMPRESORA SEIKOSHA SP 800/1000	900 ptas.
FC14	FUNDA IMPRESORA AMSTRAD DMP-1	1.205 ptas.
FC15	FUNDA IMPRESORA RITEMAN F+/C+	825 ptas.
FC16	FUNDA IMPRESORA RITEMAN 10/120	785 ptas.

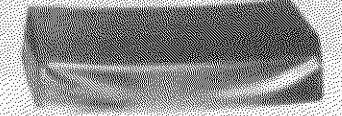


AMSTRAD CPC-6128

Si no estás interesado en alguno de estos artículos, escribenos e indicanos que accesorios necesitas para tu AMSTRAD.



IMPRESORA RITEMAN F+/C+



IMPRESORA SEIKOSHA SP-1000/800

NOTA: Disponemos de una gran variedad de fundas para otros micros e impresoras.

CONDICIONES ESPECIALES PARA DISTRIBUIDORES

- Realice su pedido por teléfono o por carta. Indiquen siempre el codigo del articulo deseado y la cantidad que desean.
- Todos los envíos se realizan contra reembolso más 160 ptas. de gastos de envío. **Todos estos precios llevan incluido el I.V.A.**

CUPON DE PEDIDO

Recorte o copie este cupón y envíelo hoy mismo a **MICROSOFT-HARD, S. L.**

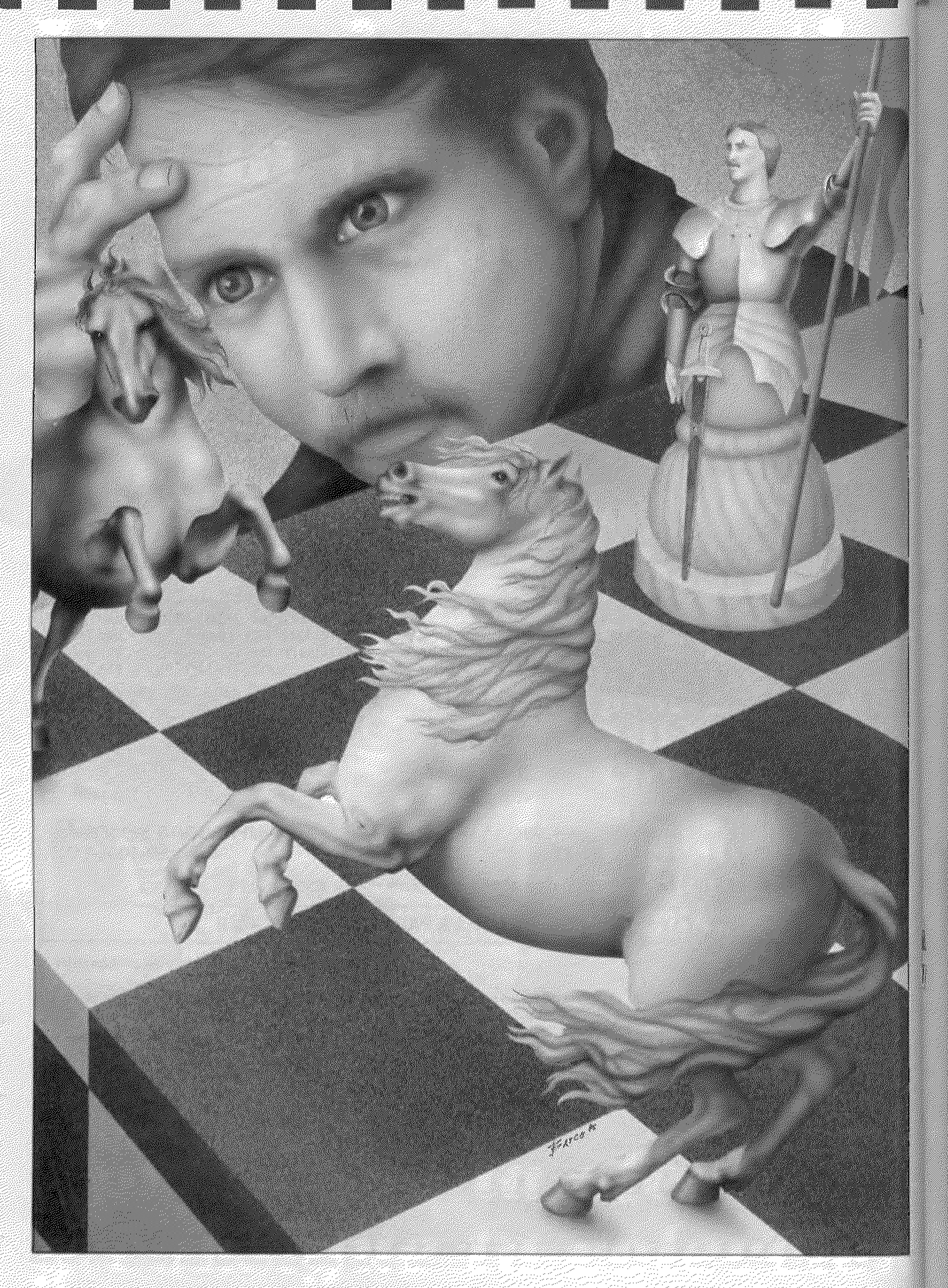
NOMBRE Y APELLIDOS ____________________

DOMICILIO ____________________

LOCALIDAD ____________________ CODIGO POSTAL ____________________

PROVINCIA ____________________ TELEFONO ____________________

Codigo-Cantidad ______ Codigo-Cantidad ______ Codigo-Cantidad ______ Codigo-Cantidad ______



ASI SE HACE UN PROGRAMA DE AJEDREZ

Luis E. JUAN VIDALES

En el mercado existen numerosos programas que juegan al ajedrez. Todos los aficionados o curiosos por el juego los han usado alguna que otra vez. Este artículo explica paso a paso cómo se hace un programa de ajedrez. Los métodos de búsqueda, las funciones de evaluación, todo.

La realización de una máquina capaz de jugar al ajedrez ha sido una meta que ha atraído la atención de los hombres durante siglos.

Ya en 1796, el barón Von Kempelen presentaba en Austria un sorprendente autómata capaz de vencer a sus contrincantes humanos, entre los que se contaron Napoleón Bonaparte y Edgar Allan Poe. Muchos años después se comprobó que **«El Turco»** *(que así se llamaba el autómata)* albergaba en su interior a un experto jugador humano de pequeña estatura, que era quien en realidad gobernaba los mecanismos.

La primera máquina jugadora de ajedrez **«sin trampa»** fue construida hacia 1890 por nuestro Leonardo Torres Quevedo y todavía existe y funciona. Se trata de un ingenio electromecánico capaz de dar mate de torre y rey contra rey.

Sin embargo, el auténtico **«padre»** del moderno ajedrez por ordenador es el matemático norteamericano Claude Elwood Shannon, quien con su artículo «Programming a computer to play chess», publicado en Scientific American en febrero de 1950, estableció los principios fundamentales que todavía constituyen la base de casi todos los programas de ajedrez para ordenador, entre ellos las famosas **«Estrategias A y B»** que se verán más adelante.

El objetivo del presente artículo es exponer las ideas y elementos básicos necesarios para la implementación de un programa de ajedrez y nuestra secreta ilusión, que algunos de nuestros lectores se sientan suficientemente animados como para intentar llevarlos a la práctica. A éstos les auguramos interminables horas de experimentación absorbente e instructiva.

ELEMENTOS DEL JUEGO

El primer obstáculo a superar cuando se pretende programar un ordenador para jugar al ajedrez es, obviamente, el de cómo «explicarle» en qué consiste el juego.

Dado que el ordenador es un ente capaz solamente de almacenar y manipular números *(a velocidad vertiginosa, desde luego)*, no sólo es necesario convertir la realidad física de un tablero de 64 casillas y 32 piezas de 12 tipos diferentes en una estructura matemática fácilmente procesable por el ordenador, sino también trasladar sus complejas reglas a funciones numéricas y, todavía más importante, reducir a ecuaciones todo el conocimiento táctico y estratégico que marca la diferencia entre un auténtico jugador de ajedrez y alguien que sólo sabe mover las piezas.

Así pues, los elementos que definen las tareas a realizar por todo programa de ajedrez son los siguientes:

a) *Representación de posiciones.*
b) *Generación de movimientos.*
c) *Evaluación de posiciones.*
d) *Estrategia de búsqueda.*

En los siguientes apartados estudiaremos la implementación de los elementos enunciados, haciendo observar que, mientras los dos primeros están ya plenamente conseguidos sobre un ordenador, no existe *(todavía)* un algoritmo óptimo para los dos últimos, siendo precisamente la evaluación de posiciones la que determina la potencia de juego del programa y la estrategia de búsqueda, su eficiencia en términos de economía de tiempo.

Hemos excluido el tratamiento de presentación de datos, que puede ir desde el simple código algebraico hasta los graficos tridimensionales, pasando por coordenadas a base de LED's, por ser un tema secundario e independiente del núcleo del programa.

REPRESENTACION DE POSICIONES

Aunque la representación numérica más inmediata de un tablero de 8×8 casillas sería una matriz de 8×8 elementos, en la práctica, y debido a las consideraciones que haremos a continuación, se prefiere utilizar un tablero de 10×12, representado por una matriz unidimensional de 120 elementos *(ver Figura 1)*.

En efecto, mediante la disposición en 10×12 se puede establecer un «borde» en torno al tablero de 8×8 a base de casillas en las que se almacena un código especial, que permite la detección inmediata de cuándo una pieza se **«sale»** del tablero. El utilizar 10×12 y no 10×10 se debe al particular movimiento del caballo, que salta 2 casillas de una vez. *(El hecho de no ser necesario un tablero de 12×12 lo dejamos a la curiosidad del lector)*. Finalmente, el empleo de una matriz unidimensional se debe a la estructura lineal de la memoria del ordenador y a la circunstancia de que éste no necesita, como nosotros, representar **«visualmente»** un tablero cuadra-

do. No obstante, en los diagramas que se incluyen en estos artículos se representa la parte central de dicha matriz en forma de 8×8 para facilitar su comprensión, quedando sobreentendida su verdadera representación en la memoria del ordenador.

[illegible]	[illegible]	[illegible]
0	Cuadro vacío	0
1	Peón	—1
2	Peón recién movido dos cuadros	—2
3	Caballo	—3
4	Alfil	—4
5	Torre no movido	—5
6	Torre movida	—6
7	Dama	—7
8	Rey no movido	—8
9	Rey movido	—9
255	Cuadro exterior al tablero	255

TABLA 1

En cuanto a las piezas, existen numerosos sistemas de representación posibles, si bien nosotros preferimos el de la Tabla 1, ya que simplifica la aplicación de las reglas del enroque y de la captura **«al paso»** de los peones. A la izquierda se representan los códigos de las piezas del programa y a la derecha, los de las piezas del contrincante. Obsérvese que el código de representación de las piezas no equivale al valor de dicha pieza en la evaluación de material.

GENERACION DE MOVIMIENTOS

Las funciones generadoras de movimientos deberán ser capaces de calcular, para cada pieza, la lista de movimientos válidos o lícitos, es decir, aquellos de los posibles que no contradigan las reglas del ajedrez. Por ejemplo, supongamos al rey blanco enrocado en la casilla 27 de la Figura 1. Sus movimientos posibles serán a las casillas 16, 17, 18, 26, 28, 36, 37 y 38. Ahora bien, las casillas 16, 17 y 18 contienen el código de «borde» y en las 26, 37 y 38 hay una torre y dos peones, respectivamente. Si el rey se ve atacado por un alfil negro en la casilla 36, sus únicos movimientos lícitos serán 36 *(capturando al alfil)* o 28.

En el mismo ejemplo, podría suceder que el alfil negro estuviera defendido por, digamos, un caballo negro. En este caso, al estudiar el programa la continuación correspondiente al movimiento 27-36, hallaría que una de las respuestas posibles de las negras sería capturar al rey blanco, por lo que, **«a posteriori»** en este caso, dicho movimiento también sería considerado como ilícito.

La forma más simple que pueden adoptar las funciones generadoras de movimientos es la de tablas de incrementos relativos a la casilla *(ver Tabla 2)* complementadas por una serie de condicionales y verificadores de la licitud de los movimientos. Véase cómo la mayor complejidad de movimientos corresponde al peón y no al caballo, debido fundamentalmente a su modo de capturar *(tanto normal como al paso)*, diferente del de las demás piezas; asimismo, la función generadora deberá incluir la promoción del peón que llega a la octava

	110	111	112	113	114	115	116	117	118	119
	100	101	102	103	104	105	106	107	108	109
8	90	91	92	93	94	95	96	97	98	99
7	80	81	82	83	84	85	86	87	88	89
6	70	71	72	73	74	75	76	77	78	79
5	60	61	62	63	64	65	66	67	68	69
4	50	51	52	53	54	55	56	57	58	59
3	40	41	42	43	44	45	46	47	48	49
2	30	31	32	33	34	35	36	37	38	39
1	20	21	22	23	24	25	26	27	28	29
	10	11	12	13	14	15	16	17	18	19
	00	01	02	03	04	05	06	07	08	09
		a	b	c	d	e	f	g	h	

FIGURA 1

Pieza	Cód.	Desliz.	Movimientos									
Peón B	1	0	+10	+20	+9	+11	+9	+11	0	0	0	0
Peón N	1	0	−10	−20	−9	−11	−9	−11	0	0	0	0
Caballo	3	0	+8	−8	+12	−12	+19	−19	+21	−21	0	0
Alfil	4	1	+9	−9	+11	−11	0	0	0	0	0	0
Torre	5	1	+1	−1	+10	−10	0	0	0	0	0	0
Torre	6	1	+1	−1	+10	−10	0	0	0	0	0	0
Dama	7	1	+1	−1	+9	−9	+10	−10	+11	−11	0	0
Rey	8	0	+1	−1	+9	−9	+10	−10	+11	−11	+2	−2
Rey	9	0	+1	−1	+9	−9	+10	−10	+11	−11	0	0

TABLA 2A

Condiciones									
1	2	4	4	5	6	0	0	0	0
1	3	4	4	5	6	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	7	8
0	0	0	0	0	0	0	0	0	0

TABLA 2B

fila. En cuanto a las piezas llamadas **«deslizantes»** *(dama, torre y alfil)*, un flag adicional indica al programa que puede continuar explorando una determinada dirección incremental mientras siga encontrando casillas vacías.

En la Tabla 2b, las condiciones asociadas a los movimientos respectivos de la Tabla 2a son las siguientes:

0 = *No hay condición.*
1 = *Si la casilla destino está vacía.*
2 = *Sólo desde fila 3 (ver Figura 1).*
3 = *Sólo desde fila 8.*
4 = *Si en casilla destino hay una pieza contraria.*
5 = *Si en casilla origen -1 hay un peón contario código 2.*
6 = *Si en casilla origen +1 hay un peón contrario código 2.*
7 = *Reglas enroque corto.*
8 = *Reglas enroque largo.*

Nótese que el enroque está considerado como movimiento de rey.

EVALUACION DE POSICIONES

Un programa que desarrollara exclusivamente los dos elementos anteriormente expuestos se limitaría a mover *(muy correctamente, eso sí)* las piezas sin ningún tipo de plan estratégico, ni siquiera táctico, ignorando completamente el fin último del ajedrez, que es el de dar mate al rey contrario.

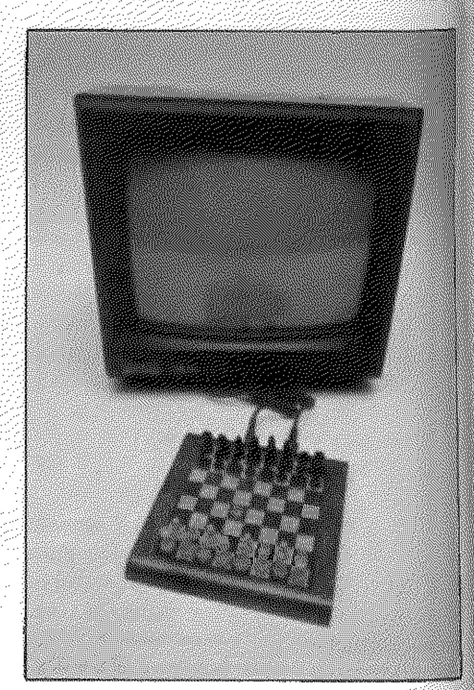
Esta regla fundamental, **«dar mate al rey contrario»**, tan fácil de enunciar, es extraordinariamente difícil de hacérsela comprender a un ordenador, hasta el punto de que sólo son posibles aproximaciones indirectas basadas en los dos principios siguientes:

a) Una mejora progresiva de la posición de las piezas propias respecto a las del contrario acabará conduciendo al mate.

b) El contrario siempre intentará mantener la posición de sus piezas tan buena como le sea posible.

El desarrollo del primer principio requiere unas funciones de evaluación de la posición, y el del segundo, unas estrategias de búsqueda.

La fuerza de juego de un programa de ajedrez viene determinada fundamentalmente

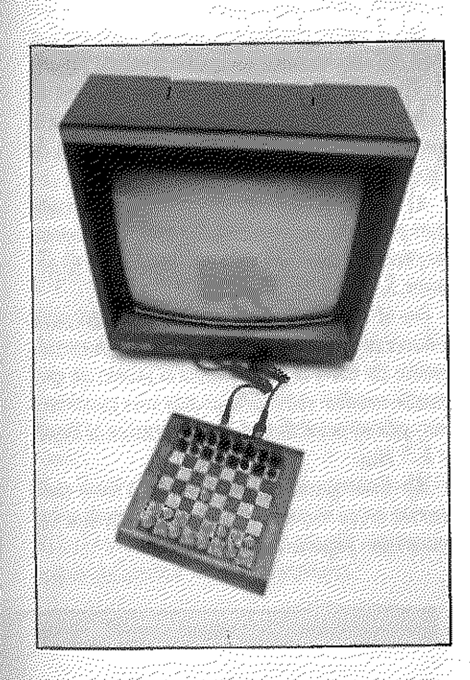


por la calidad de su función evaluadora de posiciones. Esta suele adoptar la forma siguiente:

$$F = k_1 f_1 + k_2 f_2 + \ldots + k_n f_n$$

donde f_1, f_2 ... f_n son las funciones parciales que evalúan diversos factores de la posición *(material, movilidad, estructura de peones, etc.)* y k_1, k_2 ... k_n son los coeficientes de ponderación relativa de cada factor, que pueden variar según la fase del juego en que se encuentre la partida *(apertura, medio juego o final)*.

La misión de la función evaluadora es asignar un valor numérico único a cada posición originada por el generador de movimientos, que representa la «bondad» de dicha posición *(tanto en valor absoluto como en signo)*, y en base al cual la estrategia de búsqueda decidirá el movimiento a realizar en cada turno de juego. Evidentemente, cuanto mayor sea el número *(n)* de factores considerados, más precisa será la función F, pero mayor será el tiempo de proceso requerido.



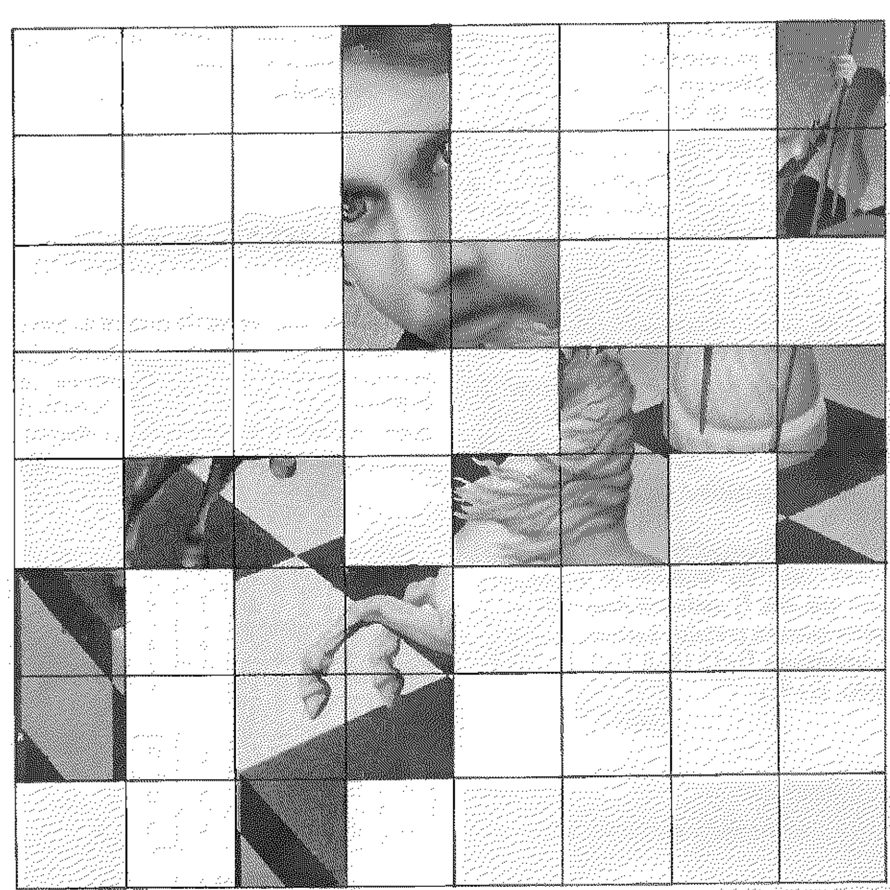
A continuación se exponen las principales funciones parciales de evaluación de la posición, f_1.

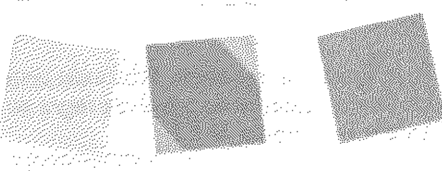
MATERIAL

Es la característica más importante del ajedrez y está constituido por las piezas presentes en el tablero por parte de cada bando, cada una con su valor relativo. El valor asignado generalmente a los distintos tipos de piezas es:

PEON = 1	TORRE = 5
CABALLO = 3	DAMA = 9
ALFIL = 3	REY = 200 (en realidad, infinito)

La función adopta simplemente la fórmula:
$f_1 = (\Sigma$ *Valores piezas propias*$) - (\Sigma$ *Valores piezas contrarias*$)$.





1	2	3	4	4	3	2	1
2	5	6	7	7	6	5	2
3	6	8	9	9	8	6	3
4	7	9	10	10	9	7	4
4	7	9	10	10	9	7	4
3	6	8	9	9	8	6	3
2	5	6	7	7	6	5	2
1	2	3	4	4	3	2	1

FIGURA 2

Inicialmente, por tanto, $f_1 = 239 - 239 = 0$.

Se puede afirmar la valoración de las piezas asignando, por ejemplo, un 3.5 a los alfiles cuando está la pareja del color en el tablero o a los caballos en las posiciones cerradas.

Finalmente, diremos que con esta única función el programa comienza ya a comportarse como un jugador, si bien su juego resulta un tanto primitivo.

MOVILIDAD

Ocupa el segundo lugar en importancia y consiste en el número total de movimientos lícitos «a priori» que el programa puede realizar con sus piezas. Como ejemplo, obsérvese que desde la posición inicial de partida cada bando tiene una movilidad = 20 (2×8 movimientos lícitos del peón más 2×2 de caballo). La función puede adoptar la forma siguiente:

$$f_2 = \frac{\text{(Mov. p. propias)} - \text{(Mov. p. contrarias)}}{\text{(Mov. p. propias)} + \text{(Mov. p. contrarias)}}$$

$$\text{Inicialmente, } f_2 = \frac{20-20}{20+20} = 0$$

DESARROLLO

Se trata de un concepto fácil de comprender pero difícil de cuantificar. Grosso modo, las piezas están desarrolladas cuando se hallan fuera de sus cuadros iniciales; no obstante, la buena práctica del ajedrez aconseja desarrollar primero caballos y alfiles, enrocar a continuación y, finalmente, desarrollar torres y dama. Asimismo, por regla general el enroque es más importante cuando la dama contraria se encuentra todavía en el tablero, debido a su peligrosa acción potencial sobre un rey ubicado en el centro.

Son muchas las funciones de evaluación posibles y aún más las excepciones a las reglas enunciadas más arriba. A título de ejemplo, presentamos la función propuesta por David Levy:

$$f_3 = D/3 - U/4 - C.E.$$

siendo:

D = N.º de caballos y alfiles propios fuera de sus cuadros iniciales *(incluyendo los que han sido capturados)*.

U = 0 si la dama propia no se ha movido o ha sido ya capturada.
= N.º de piezas propias no desarrolladas, en el resto de los casos.

C = 2 si la dama contraria está en el tablero
= *(1—P/4)* en caso negativo, siendo P el n.º de torres, caballos y alfiles contrarios capturados hasta ese momento.

E = O si el programa ya ha enrocado.
= 1/4 si el programa no ha enrocado, pero podría hacerlo por cualquiera de los lados.
= 1/3 si ya no es posible el enroque largo.
= 2/3 si ya no es posible el enroque corto.
= 1 si se ha perdido el derecho a enrocar.

A pesar de su aparente arbitrariedad, se puede comprobar mediante un análisis detenido que esta f_3 evalúa el desarrollo de las piezas con una precisión bastante aceptable.

Incialmente, D = O, U = 0, C = 2, E = 1/4. Por tanto, $f_3 = 0.5$

DOMINIO DEL CENTRO

El dominio del centro es uno de los principales objetivos del desarrollo de piezas, dado que desde las casillas centrales una pieza puede atacar más cuadros que desde los bordes y, por supuesto, más que desde las esquinas. Para evaluar esta característica, la función correspondiente posee una tabla de puntuación asociada a cada casilla *(ver figura 2)*, efectuando la operación siguiente:

$f_4 = \Sigma$ *Puntos asociados a casillas atacadas por piezas propias).*

Cabe la posibilidad de hacer más agresivo el programa desplazando el «centro» de la tabla de puntuación hacia las filas contrarias e incrementando el valor de las casillas del oponente, tal como se muestra en la figura 3, referida en este caso a las blancas.

(Nota: f_4 tiene en cuenta las casillas ***atacadas*** *debido a que, en principio, toda casilla atacada es* ***«ocupable».*** *Una función alternativa más simple se puede obtener sumando directamente los puntos asociados a las casillas* ***ocupadas*** *por piezas propias).*

9	10	11	12	12	11	10	9
9	12	13	14	14	13	12	9
9	12	14	15	15	14	12	9
9	12	14	15	15	14	12	9
8	11	13	14	14	13	11	8
6	9	11	12	12	11	9	6
4	7	8	9	9	8	7	4
2	3	4	5	5	4	3	2

FIGURA 3

ATAQUE AL REY

El método más simple de evaluar la fuerza de nuestro ataque sobre el rey contrario se basa en la misma idea de la función anterior, con la diferencia de que, en este caso, la tabla de puntos asociados a cada casilla es dinámica y está referida a la posición del rey contrario.

0	1	2	4	8	8	8	4
0	1	2	4	8	8	8	4
0	1	2	4	4	4	4	4
0	1	2	2	2	2	2	2
0	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

FIGURA 4

En la figura 4 se representa dicha tabla para las blancas, suponiendo que el rey negro estuviera en la casilla *96 (f_8).*

$f_5 = (\Sigma$ *Puntos asociados a casillas atacadas por piezas propias).*

SEGURIDAD DEL REY

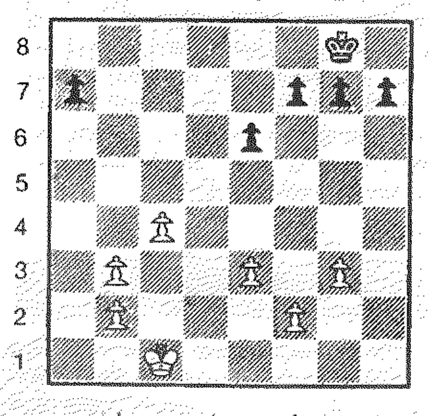
Un rey está mejor defendido cuanto menor sea el valor de las piezas defensoras y más próximas estén de aquél estas últimas. Asimismo, una pieza defendida es aún mejor defensora del rey. Estas consideraciones pueden materializarse de la forma siguiente:

$$f_6 = \Sigma_i \frac{Q_i}{V_i} (N_i + 1)$$

siendo:

Q_i = Proximidad de la pieza defensora i al rey propio según los balores de la tabla de «ataque al rey» (es decir, 8 para casillas inmediatas, 4 para la capa siguiente, etc.), aplicada al rey propio.
V_i = Valor material de la pieza defensora i.
N_i = N.º de piezas propias, incluido el rey, que defienden a su vez a la pieza defensora i.

FIGURA 5



ESTRUCTURA DE PEONES

La estructura de peones constituye el «esqueleto» de la posición y suele ser decisiva cuando se alcanza la fase final de la partida. Existen tres tipos de peones que influyen negativamente en la estructura:

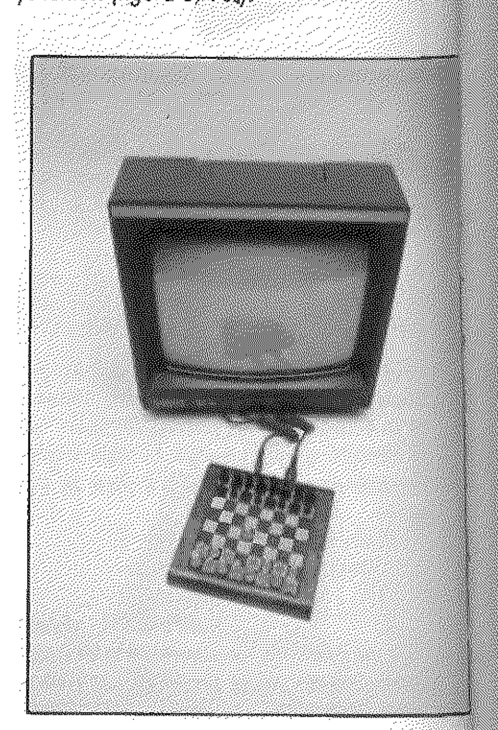
a) Peones doblados, cuando existen 2 o más del mismo color en una columna. *(figura 5, Pb_2 y Pb_3).*

b) Peón aislado, si no hay otros de su color en las columnas adyacentes. *(figura 5, Pa_7).*

c) Peón rezagado, cuando no existe ningún peón de su mismo color en las columnas adyacentes que esté más retrasado que él. *(figura 5, Pf_3).*

Por el contrario, hay un tipo de peón cuya influencia en la estructura es positiva, especialmente de cara al final *(hasta el punto de que puede resultar más valioso que una pieza mayor, por su posibilidad de promoción al llegar a 8.ª fila):*

d) Peón pasado, aquél que no tiene contrarios en su misma columna ni en las dos adyacentes. *(figura 5, Pc_4).*



La función que evalúa la estructura de peones para las blancas puede ser del tipo de:

$$f_7 = \frac{20}{M+1} \Sigma_1 G_i - N_R - 4\ N_A$$

siendo:

M = Material total en el tablero, excluidos ambos reyes.

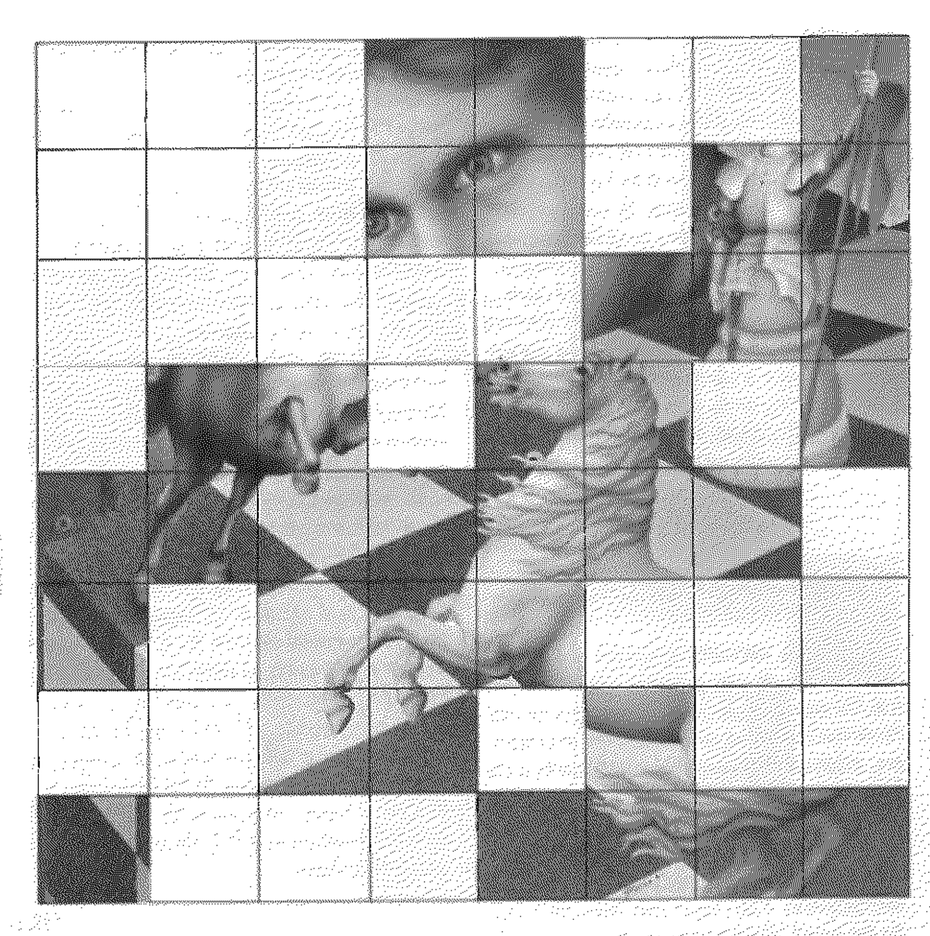
G_i = Fila ocupada *(de 2 a 7)* por el peón pasado i de las blancas.

N_R = N.° de peones blancos rezagados.

N_D = N.° de peones blancos doblados.

N_A = N.° de peones blancos aislados.

En la figura 5, $f_7 = -0.33$ para las blancas y $f_7 = -4$ para las negras.



ATAQUE Y DEFENSA DE PIEZAS

Las funciones analizadas hasta este momento realizan una evaluación de las características «estáticas» de una posición. Sin embargo, es muy interesante que el programa disponga de una función capaz de estimar de alguna manera las cualidades tácticas de la posición en términos de posibilidad de ganancia de material o, incluso, de mate al rey contrario. Dicha función podría formularse en los términos siguientes:

$$f_8 = \Sigma_i \frac{V_i\ \Sigma_i \frac{1}{V_i}}{\Sigma_i \frac{1}{V_i}}$$

donde:

V_i = Valor material de la pieza contraria atacada i.

V_h = Valores materiales de las piezas propias j que atacan la pieza i.

V_j = Valores materiales de la piezas contrarias h que defienden a su compañera i.

FINALES DE PARTIDA

No se trata tanto de establecer una función específica, como de detectar cuándo una partida ha entrado en dicha fase y obrar en consecuencia, modificando sutilmente el comportamiento del programa.

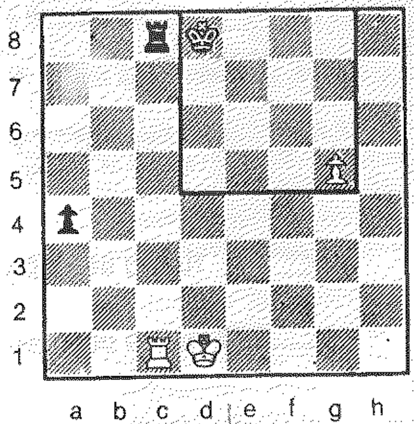
La técnica de finales es la parte más difícil de plasmar en un programa de ordenador debido a que las circunstancias más aparentemente negligibles pueden decidir el resultado final de la partida. *(Es sabido que, a igualdad de material, el hecho de que un rey esté en una casilla o en la adyacente puede invertir dicho resultado).* No obstante, sobre tal dificultad, se pueden establecer algunos criterios útiles en el tratamiento de esta fase de la partida:

a. Los coeficientes de ponderación, k, de las distintas funciones se deberán acomodar a la nueva situación. Por ejemplo, en esta fase adquieren mayor importancia el material y la estructura de peones y menor, el dominio del centro y la seguridad del rey, entre otras.

b. Los peones pasados, en particular, son de la máxima importancia, debiendo ser defendidos preferentemente por torres colocadas detrás y no debiendo ser obstaculizados por piezas propias. El programa debería bonificar y penalizar respectivamente tales defensas y obstáculos.

c. En un final de sólo peones, el rey contrario debe ser forzado, en lo posible, a salir del **«cuadrado del peón»** pasado propio. El concepto de **«cuadrado del peón»** no es exclusivo del ajedrez por computador y consiste en el cuadrado definido por el vértice *(casilla)* ocupado por el peón y la 8.ª fila, hacia el lado del rey contrario. Si dicho rey está fuera del cuadrado cuando le toca mover al peón, no podrá evitar que éste llegue a 8.ª fila. *(En la figura 6, las blancas ganan con 1.T×T+ R×T).*

FIGURA 6



d. Cuando el rey contrario ha quedado compeltamente solo, el programa debe disponer de un buen algoritmo para arrinconarlo. Se puede hacer simplemente en la misma tabla de puntuación empleada en la función **«Dominio del centro»** *(aplicada al rey contrario),* combinada con un algoritmo que procure aproximar ambos reyes. Está claro que el programa siempre elgirá el movimiento que minimice la puntuación del rey contrario, con lo cual éste será empujado progresivamente hacia una esquina. Por otra parte, la conveniencia de que el rey propio esté próximo resulta obvia.

ESTRATEGIAS DE BUSQUEDA

Según la teoría de juegos, un juego es un conjunto de reglas unívocas que determinan en todo momento el comportamiento de los jugadores. La principal característica de los juegos, y del ajedrez entre ellos, es que las pérdidas o ganancias de cada jugador no dependen sólo de sus propias acciones u omisiones,

sino también del comportamiento del contrario. Cada jugador decide su siguiente movimiento a la vista de la situación existente cuando le toca el turno, pero se puede afirmar que los jugadores tienen previsto de antemano que lo que harían en cualquiera de las circunstancias posibles. El «catálogo» de las decisiones para todas las situaciones posibles es, precisamente, lo que se denomina «estrategia».

Un juego de las características del ajedrez permite la aplicación de todo tipo de sistemas clásicos de exploración del árbol de variantes y búsqueda en grafos. En el presente artículo nos limitaremos a exponer los más típicos y representativos, sin pretender hacer una enumeración exhaustiva.

EXPLORACION DEL ARBOL

El árbol de variantes se forma al considerar, a partir de una posición dada llamada raíz, las secuencias formadas por todos los movimientos lícitos que puede efectuar el programa, todas las respuestas del contrincante a cada uno de ellos, todos los nuevos movimientos que puede realizar el programa como contrarréplica a los del contrincante, etc., hasta llegar al nivel determinado por la profundidad de búsqueda. Para tener una idea del tamaño del árbol, diremos que en una posición de ajedrez hay un promedio de 37 movimintos lícitos a disposición del jugador al que le toca mover; por lo tanto, un árbol de profundidad 4 tendrá una media de 1.900.000 variantes, equivalentes a otras tantas posiciones terminales a evaluar mediante la correspondiente función F. Dado que con profundidad 4 sólo se analizan 2 jugadas completas *(2 movimientos por parte de cada bando)*, se comprenderá la importancia de dotar al programa de una estrategia de búsqueda lo más eficiente posible.

METODO MINIMAX

En la figura 7 se representa la explosión combinatoria de un árbol de variantes de profundidad 4, relativamente «poco frondoso». La posición raíz es PO, con el turno de juego en poder del programa. Una de las posiciones terminales, por ejemplo la P1232, se alcanzaría después de la siguiente secuencia de movimientos:

Programa	Contrincante
1. M1	M12
2.M123	M1232

En las posiciones terminales el turno de juego vuelve a estar en poder del programa, indicándose a la derecha la puntuación obtenida por cada una de ellas al aplicar la función evaluadora F.

El método minimax se basa en el segundo de los principios enunciados en el apartado 5 de estos artículos; el programa «reflexionaría» de la manera siguiente:

a. La posición terminal más deseable sería la P1223, que posee la puntuación más alta de todas *(12 puntos)*.

b. Sin embargo, mi contrincante mueve 2 veces antes de que lleguemos a cualquiera de las posiciones terminales y juega *(por lo menos)* tan bien como yo, por lo cual no debo esperar que elija precisamente los movimientos que más me favorecen.

c. Así pues, debo conformarme y elegir el movimiento que conduzca a la mejor posición terminal posible suponiendo que, siempre que mueve él, elige el movimiento más perjudicial para mí.

En una búsqueda a profundidad 4, el programa deberá encontrar el máximo de los mínimos de los máximos de los mínimos, razón por la que el método recibe el nombre de «minimax». En el árbol de la figura 7, la mejor posición terminal posible es la P1231 con 5 puntos y, por tanto, el movimiento a efectuar es el M1. La variante principal es la secuencia de movimientos: M1, M12, M123, M1231.

Para llegar a tal conclusión, el programa evalúa todas y cada una de las posiciones terminales y luego «remonta» las puntuaciones hasta la posición raíz PO, de acuerdo con el siguiente criterio:

A) Si en una posición el turno de juego está en poder del contrincante, se le asigna la puntuación mínima de las posiciones derivadas.

B) Si en una posición el turno de juego está en poder del programa, se le asigna la puntuación máxima de las posiciones derivadas.

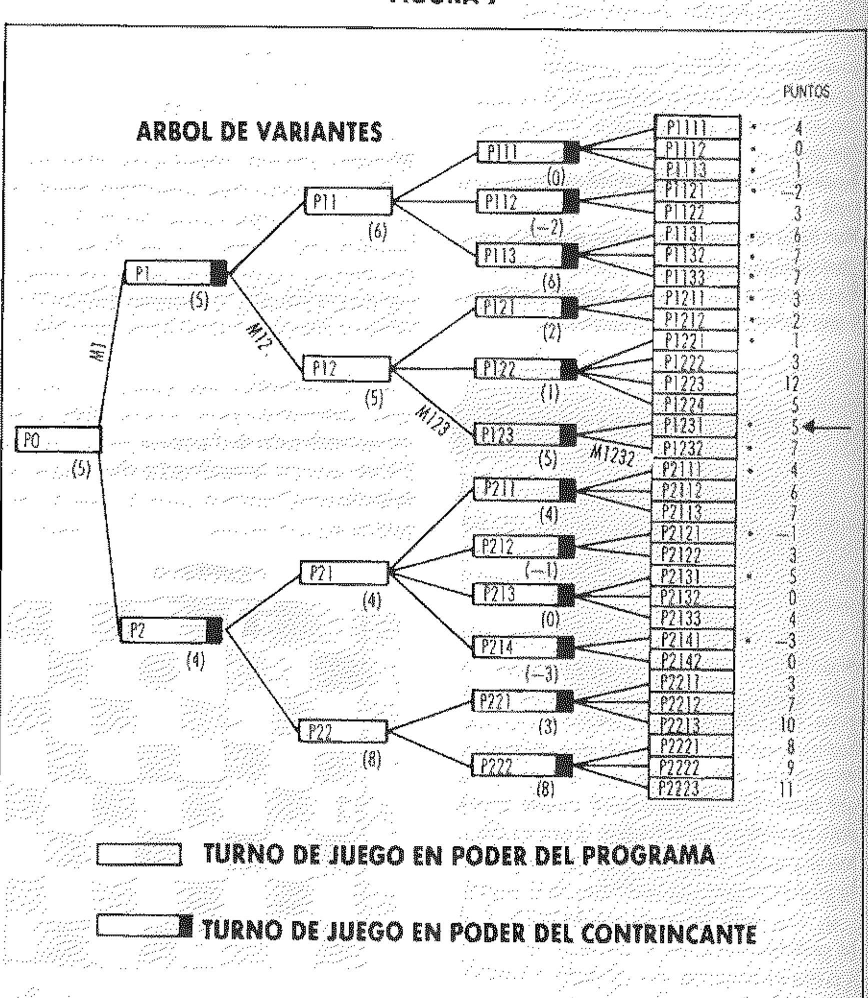
En la figura 7 se indican entre paréntesis las puntuaciones remontadas de cada posición. La posición terminal «minimax» es, obviamente, aquélla cuya puntuación «se transmite» hasta la posición raíz.

ALGORITMO ALFA-BETA

Es fácil imaginar que la aplicación directa del exhaustivo método minimax a una posición real de mitad de partida puede resultar costosísima en lo que a tiempo de proceso se refiere. Para acelerar el cómputo y mejorar su eficiencia existen diversos algoritmos que «podan» el árbol de variantes, reduciendo drásticamente el número de evaluaciones necesarias en la exploración del árbol. Uno de los más conocidos y eficientes es el llamado «algoritmo alfa-beta», que debe su nombre a la utilización de dos variables:

ALFA, que representa la mejor puntuación

FIGURA 7

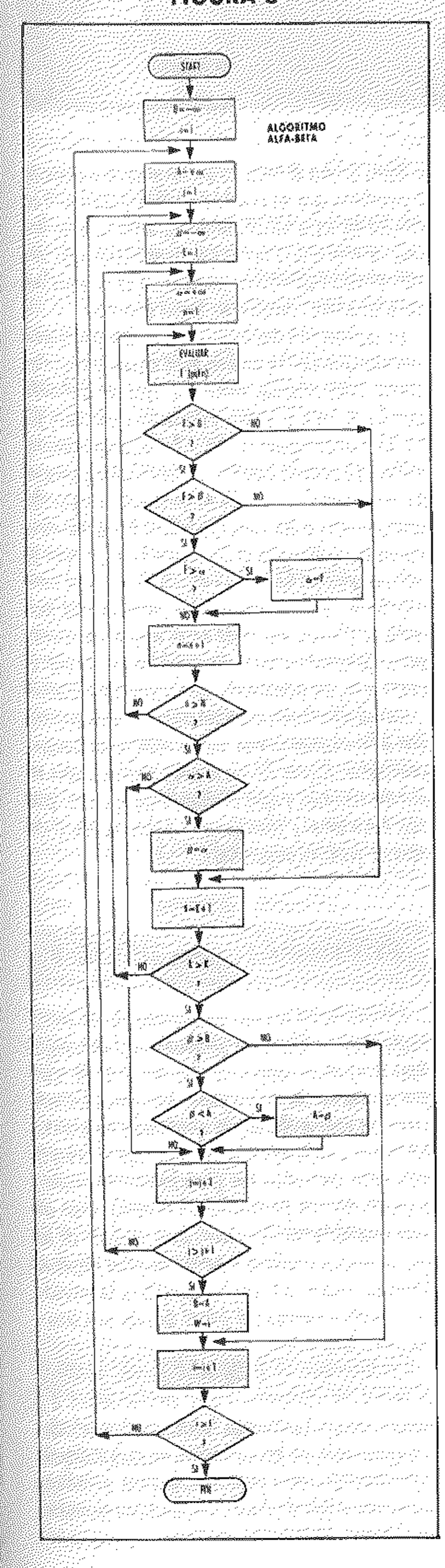


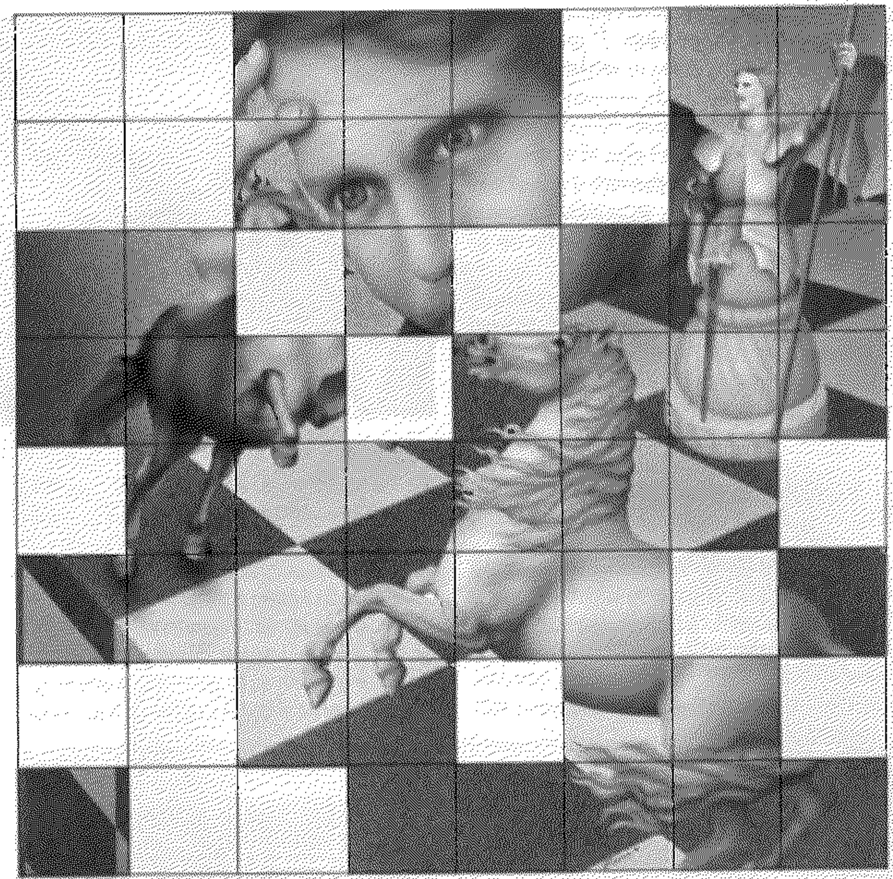
que puede conseguir el contrincante. Su valor inicial es $+\infty$ *(recuérdese que, para el contrario, es la pero puntuación posible, ya que razona al revés que el programa).*

BETA, que representa la mejor puntuación que puede lograr el programa. Su valor inicial es $-\infty$.

Alfa es el elemento «minimizante» y beta, el «maximizante». El algoritmo va, alternativamente, poniéndose en el lugar del contrincante y en el del programa y seleccionando la mejor puntuación posible para ambos bandos, rechazando sin evaluar ramas derivadas de una posición cuando la información acumulada hasta ese momento convierte en redundante su examen.

FIGURA 8





En la figura 8 se representa una implementación del algoritmo particularizado para una profundidad de 4 movimientos, por lo cual se emplean dos parejas de variables alfa-beta: *($\alpha\beta$ y (A, B)*. Los sucesivos subíndices de las posiciones terminales están representados por i, j, k, n. Asimismo, I, J, K, N, simbolizan la última rama a examinar desde la posición correspondiente. El algoritmo entrega las salidas siguientes:

B = puntuación remontada de la posición raíz, PO.

W = índice del movimiento a efectuar desde PO.

Aplicando el algoritmo alfa-beta al árbol de la figura 7, se llega exactamente a la misma conclusión que con el método minimax, pero evaluando solamente 16 posiciones terminales de las 32 que contiene el árbol *(las marcadas con *).*

Aunque este ahorro del 50 por 100 no es en absoluto despreciable, es insignificante comparado con la eficiencia que el algoritmo alfa-beta consigue cuando se aplica sobre árboles complejos *(que son los que se dan en la práctica)*. Llamando «r» al número de ramas que salen de una posición y «p» a la profundidad de búsqueda, el número de posiciones terminales a partir de una posición raíz será r^p.

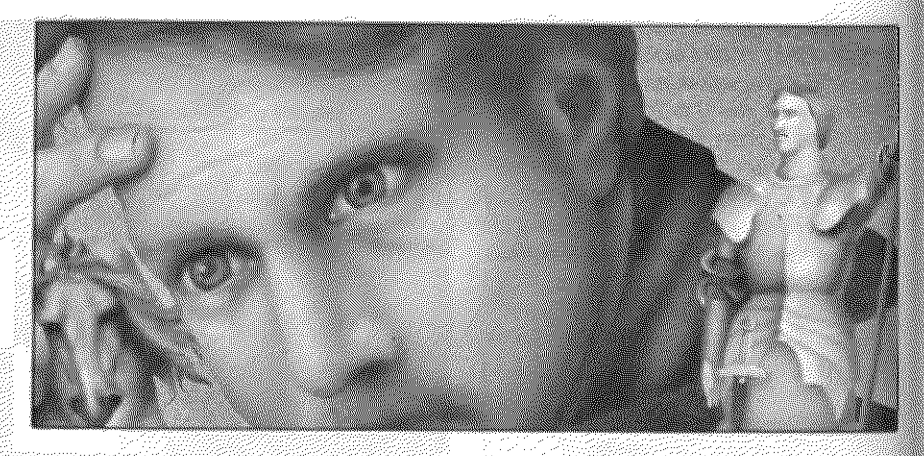
Se demuestra matemáticamente que el algoritmo alfa-beta, trabajando en condiciones óptimas (es decir, examinando en primer lugar los movimientos mejores), sólo necesita evaluar 2 r^p posiciones terminales. Esto significa que, en un árbol de $r = 37$ y $p = 4$, en el que hay 1.874.161 posiciones terminales, el algoritmo sólo examinaría 2.738, lo cual supone un ahorro de nada menos que un 99,85 por 100.

En la práctica se procura, mediante métodos empíricos, que el algoritmo trabaje en las mejores condiciones posibles, siendo típico alcanzar eficiencias superiores al 99,3 por 100.

ESTRATEGIA SHANNON TIPO A

Llamada también **«método de la fuerza bruta»**, es la estrategia de búsqueda más primitiva y se basa en la aplicación rigurosa del método minimax con o sin algoritmo alfa-beta. El árbol se forma hasta una profundidad determinada, incluyendo, por supuesto, todas las variantes y la evaluación de posiciones se realiza siempre en dicha profundidad, sin im-

portar si las posiciones terminales son **«tranquilas»** o **«turbulentas»**. El principal defecto de esta estrategia es su **«miopía»**, debida a la imposibilidad de ver más allá de la profundidad predeterminada, la cual hace que en ocasiones el programa elija el movimiento aparentemente más prometedor y con ello se sitúe al borde del abismo, especialmente cuando la posición terminal está en medio de un intercambio de piezas. No obstante, esta estrategia es la más difundida en programas de ajedrez para ordenador y sus resultados son bastante aceptables, siempre que la profundidad de búsqueda sea suficiente.



ALGORITMO DE INTERCAMBIO

Ideado para paliar los inconvenientes de la estrategia tipo A, se basa en el mantenimiento, para cada casilla de las 64 que componen el tablero, de dos listas actualizadas *(una por bando)* en las que figuran las piezas que atacan dicha casilla, ordenadas de menor a mayor valor material, o en orden de captura si éste viniera obligado por la ubicación de las piezas.

El algoritmo computa el valor de ocupar una determinada casilla o de capturar en una casilla ocupada por el contrincante, para el bando al que le toca mover. El valor de intercambio de una casilla es la ganancia material que puede esperar el bando al que le toca mover si lleva hasta el final la secuencia de capturas o cambios que es posible en dicha casilla. Por medio de este valor, el programa puede decidir si merece o no la pena realizar una determinada captura o, incluso, si la ocupación de una cierta casilla es segura o no.



La aplicación del algoritmo es simple: suponiendo que cierta casilla está ocupada por la pieza contraria de valor C_0, que es definida por C_1, C_2 y C_3, y que está bajo el ataque de las piezas propias P_1, P_2 y P_3, si el programa inicia la captura con P_1, obtiene una ganancia de material $G_1=C_0$. Si el contrario responde capturando a su vez, $G_2=C_0-P_1$. Una nueva captura llevaría a $G_3=C_0-P_1+C_1$, que se convertiría en $G_4=C_0-P_1+C_1-P_2$ si el contrincante vuelve a responder, etc. De esta manera se puede ver cuál es el mejor momento para detener la serie de capturas o si, por el contrario, es preferible no llegar a iniciarla.

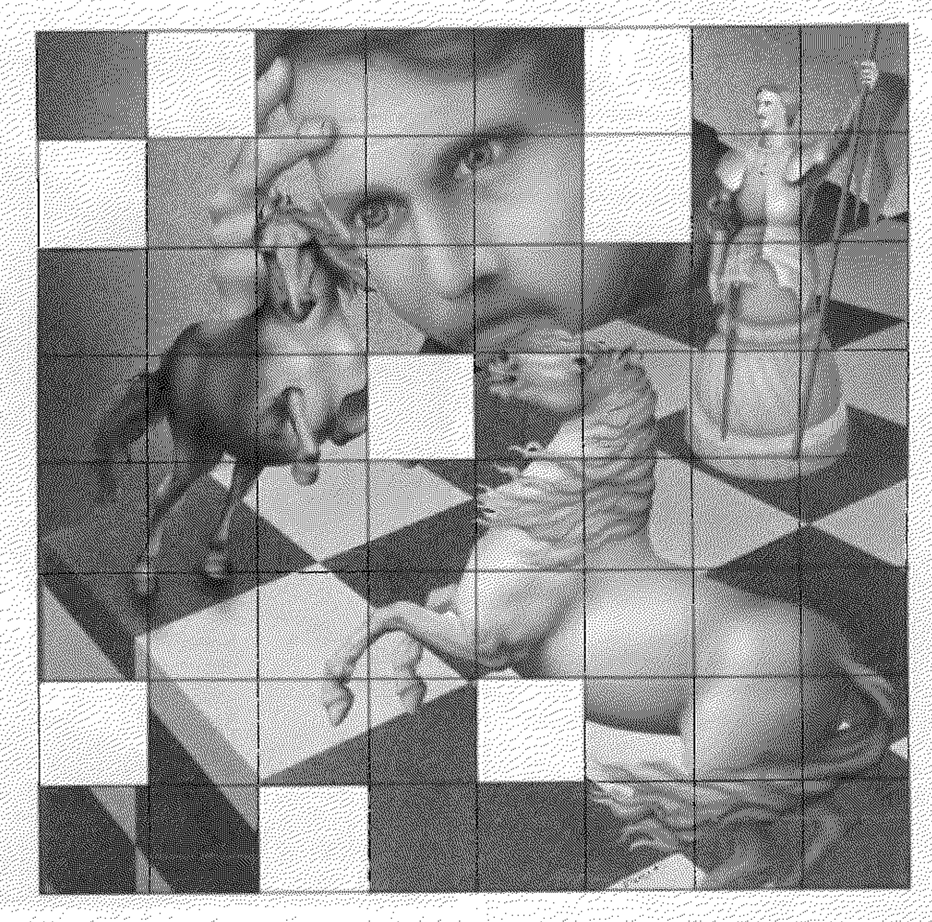
Las dos limitaciones del algoritmo son debidas a funciones extras que puede estar desempeñando alguna de las piezas involucradas en el intercambio y que el algoritmo ignora: piezas clavadas y piezas con doble función de apoyo *(sobrecargadas)*.

ESTRATEGIA SHANNON TIPO B

A la vista de los inconvenientes de la estrategia tipo A, Shannon propuso una estrategia tipo B, que se aproxima mucho más al comportamiento de un jugador humano. En este caso, la profundidad de búsqueda no está totalmente predeterminada, sino que el programa trata de todas las posiciones terminales, a partir de una profundidad mínima, sean del tipo **«estático»** *(es decir, que ninguno de los movimientos posibles a partir de ellas implique toma de material)*, prolongando la búsqueda en aquellas ramas del árbol donde sea preciso para lograrlo.

Por ejemplo, el programa puede construir el árbol de variantes hasta profundidad 2 y luego seguir buscando hasta profundidades mayores sólo en las variantes en que todos los movimientos sean capturas, jaques o réplicas a jaques, deteniéndose cuando éstas alcanzan posiciones terminales estáticas.

La base de esta estrategia reside en la mucho mayor fiabilidad de las funciones evaluadoras en las posiciones estáticas y su mayor inconveniente es el peligro de una prolongación excesiva de la búsqueda en posiciones muy «turbulentas». A este respecto, la estrategia tipo B también puede ser complementada con el algoritmo de intercambio para limitar la búsqueda.



BIBLIOTECA DE APERTURAS

Todo programa de ajedrez que se precie debe poseer una biblioteca de aperturas lo más extensa posible. Las ventajas que ello reporta son numerosas:

a. Se reduce drásticamente el tiempo de cálculo para los primeros movimientos.

b. Se aumentan las probabilidades de llegar a la fase de medio juego con una posición casi óptima a nivel teórico.

c. Se da variedad al juego frente a un contrincante humano, ya que, por su naturaleza matemática, las funciones evaluadoras y estratégicas darían siempre los mismos resultados a partir de unos mismos datos de partida.

La manera de organizar una biblioteca de aperturas es relativamente simple; puede adoptar la forma de tabla indexada, donde a cada movimiento le corresponden 2 bytes: casilla origen y casilla destino *(o casilla origen e incremento)*. Mientras el oponente siga una determinada variante, el programa se limitará a extraer su movimiento de la biblioteca. Cuando el contrincante se aparte de la línea de juego almacenada, conmutará automáticamente al método general de cálculo. En los casos en que los movimientos posibles sean múltiples para el programa *(por ejemplo, si le toca iniciar la partida)*, la selección se hará mediante un procedimiento aleatorio.

Se deberían codificar como mínimo las aperturas más importantes, incluyendo todas las variantes que la moderna teoría ajedrecística sanciona como superiores, hasta una profundidad de 4 a 10 jugadas *(dobles movimientos)*, según los casos. El volumen de memoria ocupado por una buena biblioteca de aperturas puede llegar a igualar la propia extensión del programa de cálculo, pero los beneficios obtenidos lo justifican. No hay que olvidar que la teoría de aperturas constituye una parte fundamental de los conocimientos de todo buen jugador de ajedrez.

ESTRUCTURA DEL PROGRAMA

Un programa de ajedrez para ordenador adopta básicamente la estructura del organigrama de la figura 9. Tras una primera fase en la que los movimientos son «de libro» por ambas partes, el programa comienza a generar posiciones terminales a profundidad prefijada, desde la posición raíz alcanzada en ese instante. La función evaluadora F suministra la puntuación de la posición terminal examinada a la estrategia de búsqueda, la cual va «modificando su opinión» sobre el movimiento a adoptar a medida que considera las sucesivas posiciones terminales. Es la propia estrategia, dotada normalmente de algún algoritmo de **«poda»**, la que gobierna el salto a la siguiente posición terminal a evaluar *(simbolizada por «n» en el organigrama)*.

Finalizada la búsqueda y presentado el movimiento efectuado por elprograma, caben dos posibilidades:

a. Que el programa se limite a esperar el movimiento del contrincante *(versión 4a)*.

b) Que el programa aproveche el tiempo de reflexión del contrincante para profundizar su análisis con vistas al movimiento siguiente *(versión 4b)*.

Esta última posibilidad es sumamente interesante, ya que manteniendo el mismo tiempo medio de respuesta del programa, la profundidad efectiva de análisis puede ser superior. El programa, una vez generado su movimiento, supone que la respuesta del contrincante es la que corresponde, lógicamente, a la variante principal y toma como nueva posición raíz, la resultante después de la jugada completa. *(En el ejemplo de la figura 7, una vez decidido M1 por conducir a la posición terminal P1231,* supone que la respuesta será M12 y toma como nueva posición raíz la M12). A partir de ella, y mientras no sea interrumpido por la entrada del movimiento del contrincante, lanza una nueva búsqueda anticipada, que puede llegar a ser incluso de profundidad superior si el tiempo lo permite. Si la respuesta real del contrincante no es la prevista, el programa desecha simplemente los cálculos efectuados y se enfrenta a la posición verdadera.

En la práctica, si el programa está bien diseñado, la respuesta del contrincante será la prevista en un porcentaje mayoritario de las jugadas. Llamando **«m»** a este porcentaje, podemos escribir:

$$T_r = T_p + mT_c$$

donde:

T_r = tiempo medio efectivo de reflexión disponible para el programa.

T_p = tiempo medio de respuesta del programa.

T_c = tiempo medio de respuesta del contrincante.

(T_p y T_c son los tiempos que «tarda» en emitir su movimiento el programa y su contrincante, respectivamente, tras el movimiento del otro).

Dado que $T_r > T_p$, es obvio que con un tiempo de respuesta T_p podemos conseguir una profundidad de búsqueda mayor que la que aparentemente sería posible, pues en realidad

disponemos de un tiempo T_r. *(Está claro que en la versión 4a, $T_r = T_p$).*

Es precisamente el tiempo de reflexión T_r el que determina en última instancia la fuerza de juego del programa, una vez optimizadas las funciones evaluadoras y las estrategias de búsqueda y, de hecho, es el parámetro sobre el que actúa la selección de nivel de juego en los programas.

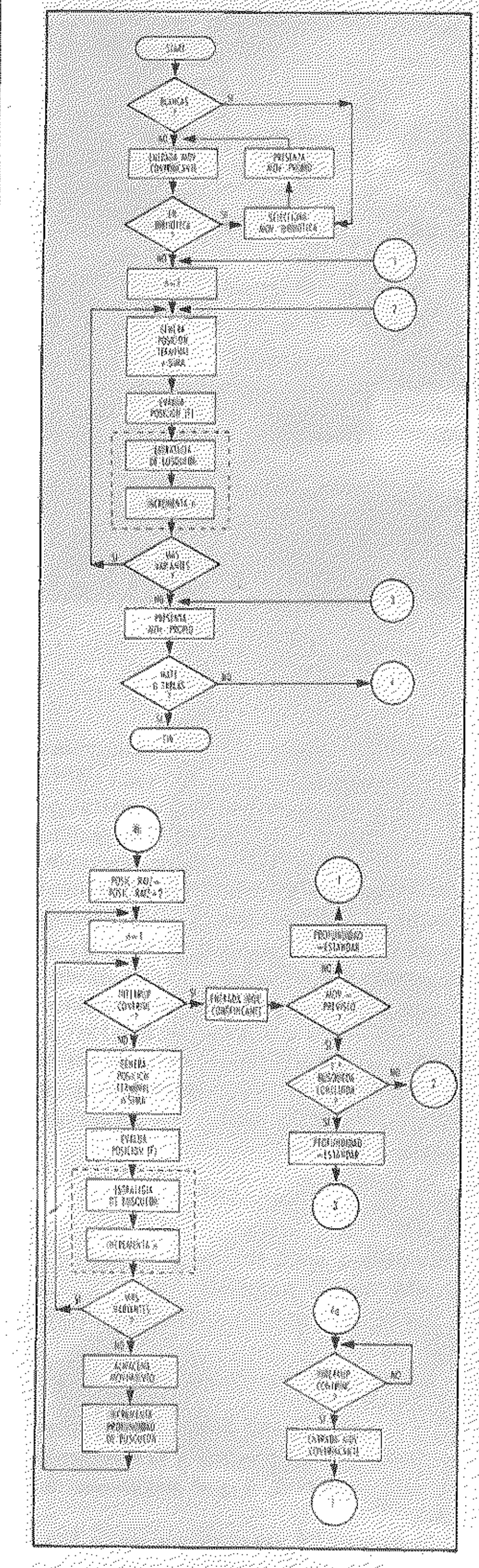


FIGURA 9
ESTRUCTURA DEL PROGRAMA



CONCLUSION

La realización de un programa de ajedrez, como esperábamos haya quedado patente a lo largo de estos artículos, es una tarea a la vez titánica y atrayente. Es titánica porque el número de rutinas diversas a desarrollar, algunas de ellas bastante complejas, es muy elevado. Y es atrayente porque la **«criatura»** simula algunos de los más interesantes procesos de la mente humana *(muchos de los conceptos involucrados en su creación forman parte de los sistemas aplicados en inteligencia artificial, como el método minimax y el algoritmo alfa-beta)* y su complejidad supone un reto para el programador.

Si alguno de nuestros lectores se siente con fuerzas suficientes para afrontar la realización de un programa de ajedrez, le haremos las sugerencias siguientes:

a. El programa debe ser lo más modular posible. Dada su extrema complejidad, es preciso facilitar las correcciones y los cambios.

b. A pesar de su lentitud, resulta de gran ayuda el realizar una versión previa, aunque sea parcial (una especie de maqueta), en Basic.

c. No conviene afrontar directamente una función de evaluación que incluya todas las parciales descritas. El programa funcionará aceptablemente con sólo incluir material, movilidad y ataque al rey, por ejemplo.

d. El valor relativo de los coeficientes k de ponderación es extremadamente importante y deberá ser determinado experimentalmente; como orientación, k_1 suele valer de 8 a 12 veces más que cualquiera de los demás coeficientes.

e. Aunque el programa resulte un tanto estúpido, no es mala idea empezar con una estrategia tipo A a profundidad 2, con minimax y sin algoritmo de intercambio.

f. Al principio, habrá que conformarse con que el programa juegue rematadamente mal los finales *(de hecho, es casi milagroso que en algún momento llegue a jugarlos verdaderamente bien).*

Finalmente, para los que deseen profundizar sobre el tema, recomendamos excelentes libros:

—**«Manual de ajedrez por computadora»**, de David Levy.
Editorial Mitre. *Barcelona, 1986.*

«Ajedrez y computadoras», de L. Pachman y V. I. Kühnmund.
Ediciones Martínez Roca. *(Colección ESCAQUES). Barcelona, 1982.*

EL MUNDO DE LOS JUEGOS

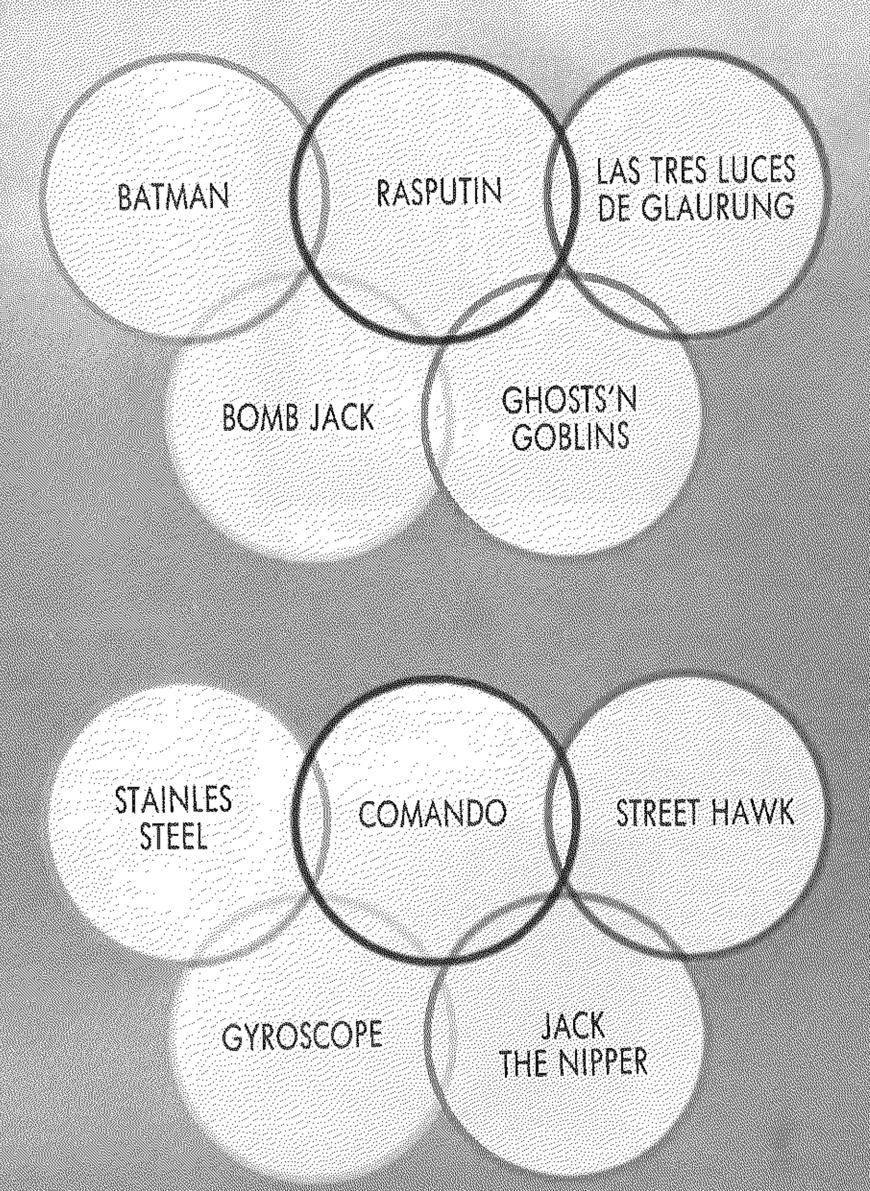
POKES

El mundo del software de juegos para Amstrad es más rico y complejo cada día que pasa. La cantidad de títulos, si los pusiésemos todos juntos, alcanzaría una altitud realmente sorprendente. Además, la gente que compra juegos padece una «obsesión» de las más lógicas del mundo: les gustaría conocerlos a fondo, recorrerlos hasta el final y, si son demasiado difíciles, encontrar una pequeña ayuda en alguna parte que les permita enterarse de la totalidad del juego en un tiempo razonable.

Más de un lector se estará imaginando a dónde queremos llegar: a los programas cargadores que permiten a los poseedores de una copia legal de un juego hacer maravillas en él.

En efecto, de eso se trata. Os presentamos 10 cargadores completos para que disfrutéis a tope de vuestros programas favoritos.

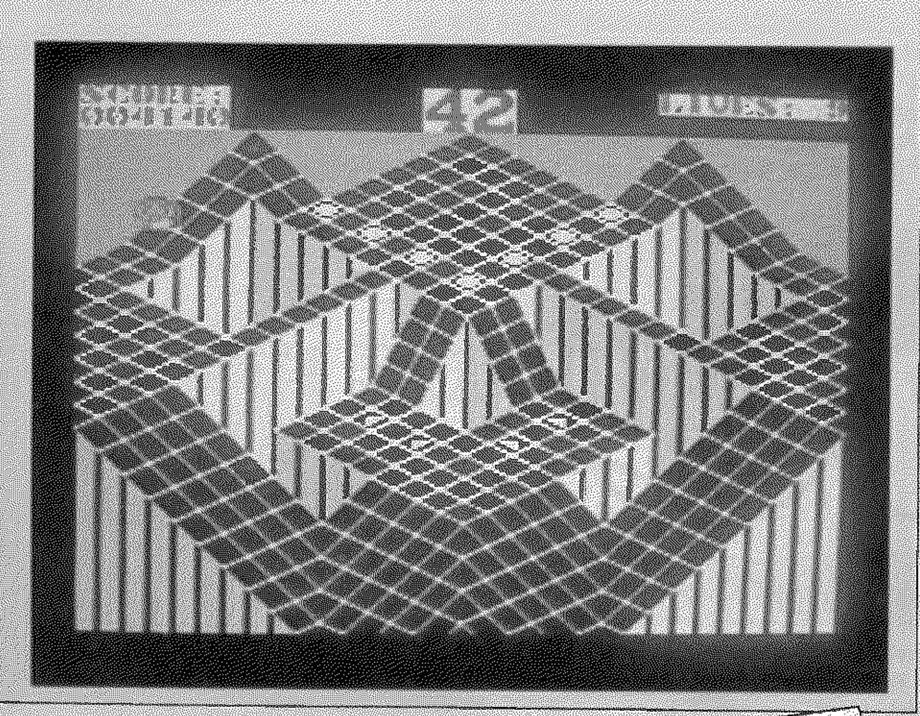
Quisiéramos dar las gracias a nuestra revista hermana, Micromanía, por proporcionarnos todo tipo de asistencia y ayuda en un campo de la informática en el que son, sin discusión, los primeros expertos.



GYROSCOPE

Estamos frente a uno de los juegos más difíciles y originales que hemos visto nunca: el arte de conducir un mecanismo giroscópico, alias peonza, por los intrincados paisajes de un laberinto tridimensional, frío y desierto como él solo, no es fácil de dominar, y justifica plenamente la inclusión de un cargador completo, cuyos secretos no vamos a descubrir por aquello de no aguar la fiesta a los fieles seguidores de la Orden del Joystick. Vidas infinitas, enemigos inocuos y mucho más os esperan detrás de unas pocas líneas de programa.

```
10 REM CARGADOR PARA GYROSCOPE
20 MEMORY &39AE:MODE 1
30 FOR F= &5000 TO &5032:READ A:POK
E F,A:NEXT
40 INPUT "VIDAS INFINITAS (S/N)";A$
:IF A$="N" OR A$="n" THEN POKE &502
2,&35
50 INPUT "JUEGO MAS FACIL (S/N)";A$
:IF A$="N" OR A$="n" THEN POKE &502
7,&3A
60 INPUT "TIEMPO INFINITO (S/N)";A$
:IF A$="N" OR A$="n" THEN POKE &502
C,&3A
70 CLS:LOCATE 12,12:PRINT"INSERTA E
L ORIGINAL":LOAD"!"
80 POKE &39E2,&C3:POKE &39E3,&9:POK
E &39E4,&50:CALL &5000
100 DATA 33,255,171,17,64,0,195
110 DATA 175,57,62,63,50,43,2
120 DATA 62,76,50,44,2,33,33
130 DATA 80,17,0,7,1,18,0
140 DATA 237,176,243,241,201,62,0
150 DATA 50,43,83,62,201,50,132
160 DATA 83,62,201,50,130,85,195,16
,79
```



STREET HAWK

Poca gente se habrá olvidado de las increíbles aventuras del protagonista de la serie televisiva «El halcón callejero».

Sus espectaculares carreras a tumba abierta con enemigos maléficos, sus increíbles saltos a través de todo tipo de obstáculos, sus certeros disparos con el láser frontal, todo está presente en un magnífico juego del mismo nombre.

Naturalmente, es bastante difícil,



y jugar con él sin la ayuda de nuestro programa cargador puede resultar harto frustante.

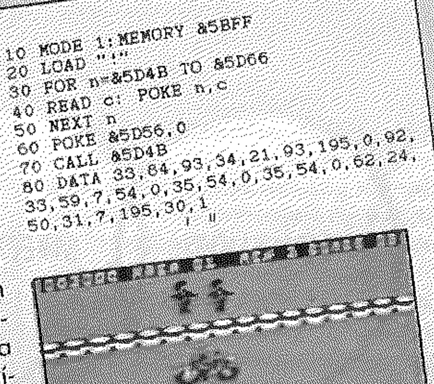
Por eso, seguro que agradeceréis el turbo infinito y los cohetes propulsores también infinitos.

Ahora a teclear, aprieta los p... ños y...

COMANDO

Es de suponer que te pasastes horas y horas delante de tu ordenador para poder llegar al final de este juego lleno de acción. Quizás lo conseguiste o quizás no. La verdad es que hay que ser un auténtico superhéroe para conseguir que no te maten la multitud de soldados enemigos que te acosan por tener la valentía de infiltrarte en sus líneas. Aquellos que no lo lograran en su momento nos hemos hecho con estos pokes de vidas infinitas. Así tú también podrás hacerte la ilusión de que tú también eres un héroe de película.

```
10 MODE 1:MEMORY &5BFF
20 LOAD "!"
30 FOR n=&5D4B TO &5D66
40 READ c: POKE n,c
50 NEXT n
60 POKE &5D56,0
70 CALL &5D4B
80 DATA 33,84,93,34,21,93,195,0,92,
33,59,7,54,0,35,54,0,35,54,0,62,24,
50,31,7,195,30,1
```



¡VIVA LA TRAVESURA! AYUDEMOS A JACK THE NIPPER

Porque, ¿quién no ha tenido ganas de tirar de la cola a un perro, arrojar objetos a la gente o ponerse a tirar petardos por la calle? Pues, seguramente toda la gente «adulta».

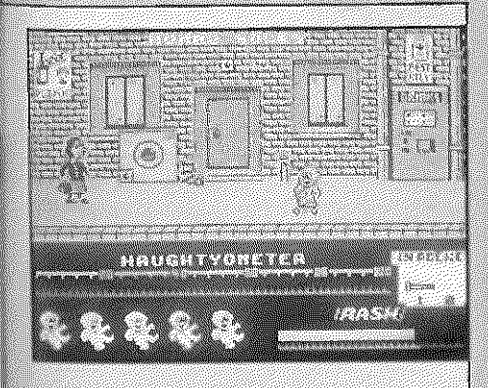
Y como las frustraciones son malas de guardar, con Jack the Nipper tendremos la oportunidad de vivir todas las cosas que siempre quisimos hacer, pero que nunca nos dejaron.

Pero como esto cansa, si no que se lo digan a los «babyes» cuando caen en la cama, aquí tenemos una ración de fuertes medicinas. Lea, lea...

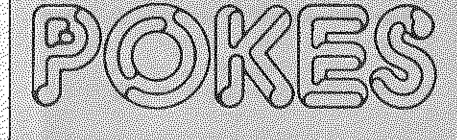
```
EM *** POR FRANCISCO SALINAS Y
         SALVADOR DE SAMPED
**
EM ***************************
***
 b=0:FOR 1=&4FD9 TO &508B:READ a:
 1,a:b=b+a:NEXT
 IF b<> 18322 THEN PRINT "ERROR E
 DATAS...":END
 INPUT"]TEMPERATURA INVARIABLE (S
 ";A$:IF A$="S" OR A$="s" THEN PO
 &5042,0
 INPUT"]TURBO INFINITO (S/N)";A$:
 A$="S" OR A$="s" THEN POKE &504A

 INPUT"]AIR JET INFINITO (S/N)";A
 IF A$="S" OR A$="s" THEN POKE &50
 ,0
 INPUT"]LASER INFINITO (S/N)";A$:
 A$="S" OR A$="s" THEN POKE &5060

 INPUT"]ARMADURA INFINITA (S/N)";
 IF A$="S" OR A$="s" THEN POKE &5
 E,0
100 CALL &4FD9
110 DATA 6,0,17,0,128,205,119
120 DATA 188,33,175,57,205,131,188
130 DATA 205,122,188,33,226,57,54
140 DATA 195,35,54,13,35,54,80
150 DATA 49,255,191,33,152,0,229
160 DATA 33,137,127,229,33,64,0
170 DATA 229,33,255,171,17,64,0
180 DATA 195,175,57,243,33,1,2
190 DATA 54,105,35,54,9,33,21
200 DATA 2,54,89,33,24,2,54
210 DATA 217,35,54,195,33,29,2
220 DATA 54,49,35,54,218,35,54
230 DATA 105,35,54,198,33,62,80
240 DATA 17,0,191,1,85,0,237
250 DATA 176,241,201,33,204,133,24
260 DATA 3,205,73,191,33,243
270 DATA 24,9,205,73,191,33,102,134
280 DATA 139,205,73,191,33,132
290 DATA 24,3,205,73,191,33
300 DATA 139,24,9,205,73,191,33,75
310 DATA 204,163,205,73,191,33
320 DATA 146,24,21,205,73,191,33,228
330 DATA 71,148,205,73,191,33,195,161
340 DATA 133,205,73,191,195,0,169,1
350 DATA 205,73,191,195,0,169,1
360 DATA 0,03,113,35,16,252,201
```




```
10 REM Jack the Nipper
20 REM
30 REM por A.C.L.
40 REM
50 LOAD"!NIPPER":POKE &A058,0:POKE
&A059,&A5
60 FOR I=&A500 TO &A50A:READ A:POKE
 I,A:NEXT
70 CALL &A000
80 DATA &3E,0,&32,&FF,&39,&32,&00,&
3A,&C3,&01,&10
```

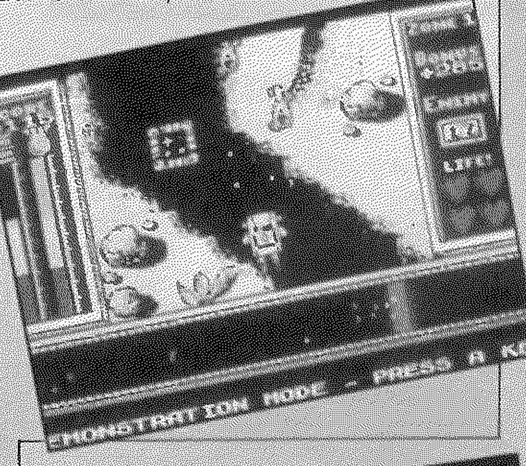


STAINLES STEEL

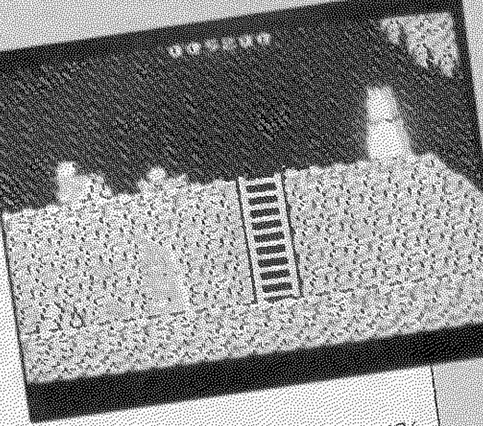
Con este curioso nombre, igual a la frase que insertada en la tapa posterior de muchos relojes, se nos presentó hace tiempo un ceñudo muchacho. Stain tenía pinta de duro, bueno, de punk duro. Más tarde descubrí que no sólo tenía la pinta de duro. Stain era un punk como los de antes, no como ésos que se conforman con llevar extraña pelambrera y beber a todas horas en la calle. A Stain se le podía mirar fijo, se enfadaba. A Stain era mejor dejarle solo.

Por eso cuando los androides invadieron la Tierra se molestó mucho. Pero no fue debido a que fueran a destruir la raza, qué va, se molestó porque no podía circular a toda velocidad por las autopistas que era lo que a él le gustaba.

Mal, muy mal lo van a pasar los androides.




```
10 MEMORY &3FFF:LOAD"!",&4000
20 MODE 2:INPUT"Vidas infinitas (S/
N) : ",a$:a$=UPPER$(a$):IF a$="S" T
HEN POKE 0,1 ELSE IF a$="N" THEN PO
KE 0,0 ELSE 20
30 POKE 1,0
40 INPUT"Fase inicial (0/1/2/3) : "
,a:IF a>3 THEN 40 ELSE IF a<0 THEN
40
50 fa=a
60 POKE &4084,&40:POKE &4085,0:sum=
0:FOR i=&40 TO 112:READ a$:a$="&"+a
$:POKE i,VAL(a$):sum=sum+PEEK(i):NE
XT
70 MODE 1:POKE &69,fa:CALL &4000
80 DATA 3A,0,0,B7,28,A,AF,32,F3,6,3
2,F4,6,32,F5,6,3A,1,0,B7,28,4,AF,32
,79,4,3E,CD,32,7D,14,21,68,0,22,7E,
14,C3,16,1,3E,fa,32,E8,BF,C9,C3,16,
1
```

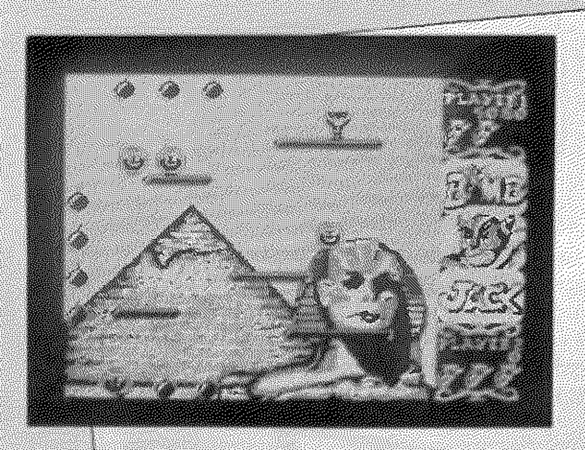


POKES DE GHOTS'N GOBLINS, O COMO TERMINAR CON UNA PESADILLA

Hemos decidido que el caballero Godofredo lleve una vida más relajada. Por eso, y porque pensamos que si él puede tomarse la vida con más calma, nosotros también, aquí os traemos los «pokecitos» necesarios. Con ellos no dudamos que nuestro valeroso paladín logrará rescatar a su amada de las garras de un funesto señor feudal.

No más stress, basta de prisas, abajo las presiones; tomémonos los paseos por los cementerios de zombies con la mayor naturalidad. Consigamos cortar las plantas carnívoras, como si de margaritas de un bucólico campillo se tratara.

```
10 MODE 2:INPUT"Vidas infinitas (S/
N) : ",a$:a$=UPPER$(a$):IF a$="S" T
HEN vi=1 ELSE IF a$="N" THEN vi=0 E
LSE 10
20 INPUT "Fase (0/2) : ",a:IF a<0 O
R a>2 THEN 20 ELSE fa=a
30 IF fa=0 THEN fa=256
40 MODE 0:BORDER 0
50 FOR i=0 TO 15:READ c:INK 1,c:NEX
T
60 LOAD"!p":MEMORY 4999:LOAD"!code"
70 IF vi THEN POKE &509B,0:POKE &50
9C,0:POKE &509D,0
80 POKE &50AC,fa-1
90 CALL &5000
100 DATA 26,13,0,15,24,6,16,2,14,1,
3,11,10,9,18,25
```





POR SIEMPRE BOMB JACK

Bueno, la verdad sea dicha, apreciamos a nuestro superratón. Así que, un día nos pidió ayuda para poder acabar con los terribles alienígenas invasores. Nosotros, acongojados por su exceso de trabajo, le ofrecimos un elixir mágico de vidas infinitas y le quitamos de enmedio las plataformas que le molestaban.

Con renovados ímpetus volvió a pelear nuestro amigo Bomb Jack. Su cara, radiante de energía y felicidad, se paseó por las Pirámides de Egipto, bonitos castillos franceses y grandes ciudades del mundo, guardándolos y protegiéndolos de las malas intenciones de los invasores.

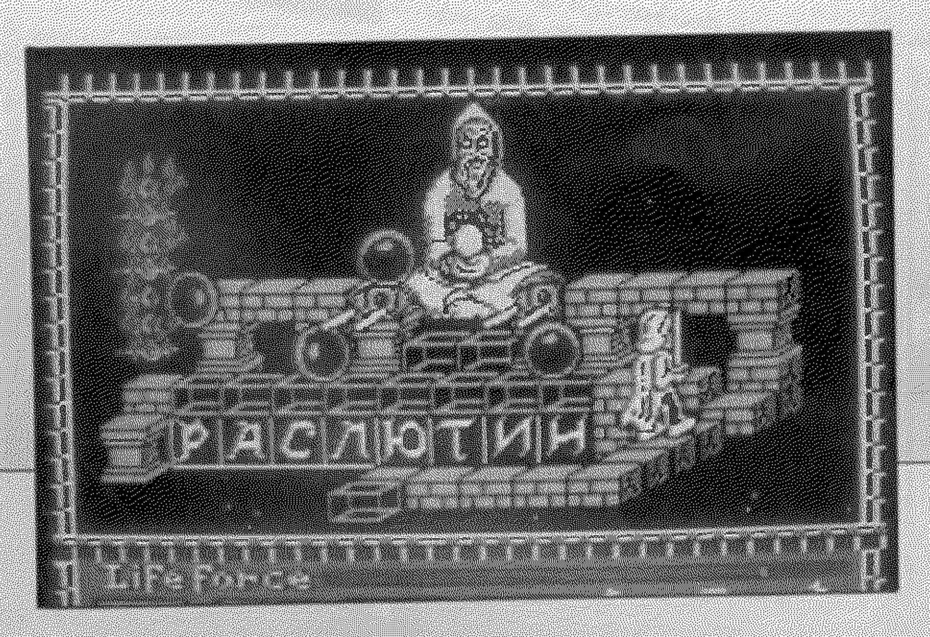
Ved, ved, la fórmula mágica:

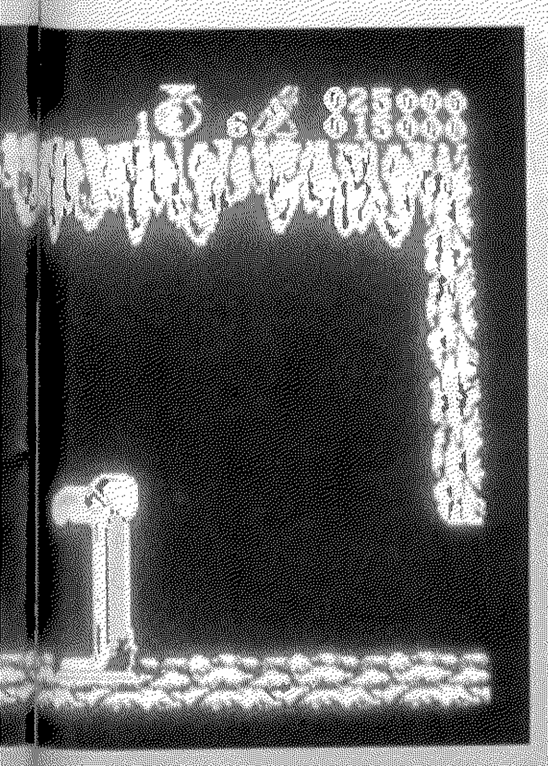
```
10 MODE 1:INPUT"Numero de vidas (1-
255) : ",a$:IF VAL(a$)<1 OR VAL(a$)
>255 THEN 10 ELSE nv=VAL(a$)
20 LOCATE 1,2:INPUT "Vidas infinita
s (S/N) : ",a$:IF LEN(a$)<>1 THEN 2
0 ELSE a$=UPPER$(a$):IF a$<>"S" AND
 a$<>"N" THEN 20 ELSE IF a$="S" THE
N vi=1
30 LOCATE 1,3:INPUT"Jugar sin decor
ados (S/N) : ",a$:IF LEN(a$)<>1 THE
N 30 ELSE a$=UPPER$(a$):IF a$<>"S"
AND a$<>"N" THEN 30 ELSE IF a$="S"
THEN de=1
40 LOCATE 1,4:INPUT"Jugar sin plata
formas (S/N) : ",a$:IF LEN(a$)<>1 T
HEN 40 ELSE a$=UPPER$(a$):IF a$<>"S
" AND a$<>"N" THEN 40 ELSE IF a$="S
" THEN pl=1
50 MEMORY 5999:BORDER 0:CLS:FOR f=0
 TO 15:INK f,0:NEXT f:MODE 0:LOAD "
!bjscreen.bin",49152:FOR f=0 TO 15:
READ a:INK f,a:NEXT f:LOAD "!bjcode
.bin",6000
60 POKE &1800,nv:POKE &1811,nv:IF v
i=1 THEN POKE &19FD,0
70 IF de=1 THEN POKE &19DF,0:POKE &
19E0,0:POKE &19E1,0
80 IF pl=1 THEN POKE &19E8,0:POKE &
19E9,0:POKE &19EA,0
90 CALL 6000
100 DATA 1,0,26,8,24,13,11,6,15,16,
5,2,6,3,20,10
```

RASPUTIN

Con estos pokes de escudos y espadas infinitas y energía inagotable podrás vencer al malvado Rasputín y apoderarte de la Joya de los Siete Planetas. Esta joya es la fuente de su inmenso poder y, para evitar que alguien se la quite, la ha protegido mediante los ocho conjuros más fuertes que conoce en el mundo atemporal e ilocalizable de los Siete Planetas. Suponemos que te serán muy útiles y lograrás tu objetivo, de no ser así la furia de sus monstruos dará cuenta de ti.

```
10 MODE 1:BORDER 2:INK 0,0:INK 1,26
:INK 2,2:INK 3,6
20 WINDOW #1,14,26,5,5:WINDOW #2,14
,26,7,7
30 MEMORY &1FFF:LOAD"rascode",&A400
40 PRINT"          INSERTA EL ORIGINA
L"
50 LOAD"!rasputin1":a=&1681
60 POKE a,&C3:POKE a+1,0:POKE a+2,&
A4
70 INPUT"espadas y escudos inagotab
les";a$:IF UPPER$(a$)="N" THEN POKE
 &A462,&39
80 INPUT"energia infinita";a$:IF UP
PER$(a$)="N" THEN POKE &A43A,&C
90 CLS
100 CALL &1600
```





LAS TRES LUCES DE GLAURUNG

POKES

Tiempo atrás, los jóvenes se ganaban la gloria matando a pérfidos magos, malvados caballeros, o rescatando a princesas prisioneras de pérfidos señores.

Nuestro rústico protagonista se quiso ganar unos dineros y un nombre a costa de salvar al mago del señor de la región.

Y para cumplir adecuadamente con esta misión, no hay nada como tomársela con calma.

Nosotros recomendaríamos la siguiente receta.

Nueve partes de vidas infinitas y una parte de flechas inagotables. Más o menos así:

DISCO:

```
10 OPENOUT"a"
20 MEMORY 999:MODE 1:BORDER 0:INK 0
,0:INK 1,25:INK 2,10:INK 3,15
30 ON ERROR GOTO 50
50 LOAD"glaurung.bin":POKE 42138,20
1
60 LOCATE 1,20:INPUT"VIDAS INFINITA
S (s/n)",a$:IF a$="s" THEN POKE &85
BF,0:POKE &85C0,0
70 LOCATE 1,20:INPUT"FLECHAS INFINI
TAS (s/n)",a$:IF a$="s" THEN POKE &
9029,0
80 CALL 41162
```

CINTA:

```
10 OPENOUT"a"
20 MEMORY 1000:MODE 1:BORDER 0:INK
0,0:INK 1,25:INK 2,10:INK 3,15:PRIN
T"INTRODUCE LA CINTA ORIGINAL"
30 ON ERROR GOTO 50
40 LOAD"!",0
50 LOAD"!":POKE 42138,201:CALL 4210
0
60 LOCATE 1,20:INPUT"VIDAS INFINITA
S (s/n)",a$:IF a$="s" THEN POKE &85
BF,0:POKE &85C0,0
70 LOCATE 1,20:INPUT"FLECHAS INFINI
TAS (s/n)",a$:IF a$="s" THEN POKE &
9029,0
80 CALL 41162
```

BATMAN

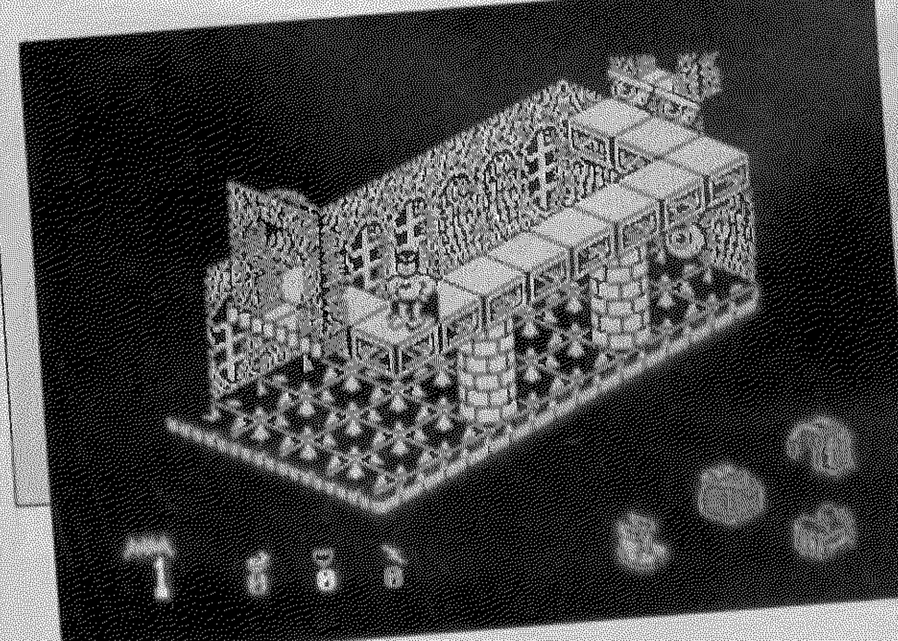
Aquí tenéis un cargador para el Batman, superhombre dotado de sentidos de murciélago, que en esta ocasión se va a enfrentar al lo-cuelo y diabólico Arlequín que le ha robado el Baticar, su especialísimo coche.

Arlequín lo escondió en su cueva, pero despiezado en siete partes. Batman debe reunirlas y montar el coche.

Lo que está claro es que su enemigo ha escondido el coche en salas llenas de peligros y trampas, por lo que Batman tendrá que luchar duro.

Pero ahora que puedes ayudarle, no dudamos que logrará cumplir su misión.

```
10 MODE 1:MEMORY &39AE:LOAD"BATMAN"
:POKE &39E4,&C3:POKE &39E5,0:POKE &
39E6,&A0
20 RESTORE:FOR I=&A000 TO &A0AC:REA
D A$:A$="&"+A$:A=VAL(A$):check=chec
k+a:POKE I,A:NEXT:IF check<>18625 T
HEN PRINT "error datas"
40 MODE 2:INPUT"Vidas infinitas : "
,a$:a$=UPPER$(a$):IF a$<>"S" AND a$
<>"N" THEN 40 ELSE IF a$="S" THEN P
OKE 0,1
50 INPUT"Tener el BATICAR montado :
",a$:a$=UPPER$(a$):IF a$<>"S" AND
a$<>"N" THEN 50 ELSE IF a$="S" THEN
 POKE 1,1
60 CALL &39AF
70 DATA CD,4B,A0,6,FF,3A,46,BE,A0,F
E,3D,28,3,5,18,F5,5,5,78,32,3E,A0,C
D,2E,A0,21
80 DATA 68,A0,11,64,0,1,9C,0,ED,B0,
11,64,0,ED,53,EC,BC,C3,BF,BC,21,41,
A0,E5,33
90 DATA 33,21,46,BB,11,46,BB,1,AC,1
,3E,9A,ED,4F,ED,5F,AE,77,ED,A0,E0,3
B,3B,E8,21,5E,A0,E5,33,33,21,40,0,1
1,0,BB,1,F2,1
100 DATA 3E,8B,ED,4F,ED,5F,AE,77,ED
,A0,E0,3B,3B,E8,3A,0,0,FE,1,20,13,3
E,99,32,EC
110 DATA 1C,DD,21,EC 1C,AF,DD,77,0,
DD,77,1,DD,77,2,3A,1,0,FE,1,C2,0,1,
AF,DD,21
120 DATA E6,1A,DD,77,0,DD,77,1,DD,7
7,2,DD,21,21,34,DD,77,0,DD,77,1,DD,
77,2,3E,7F,32,B3,31,C3,0,1
```

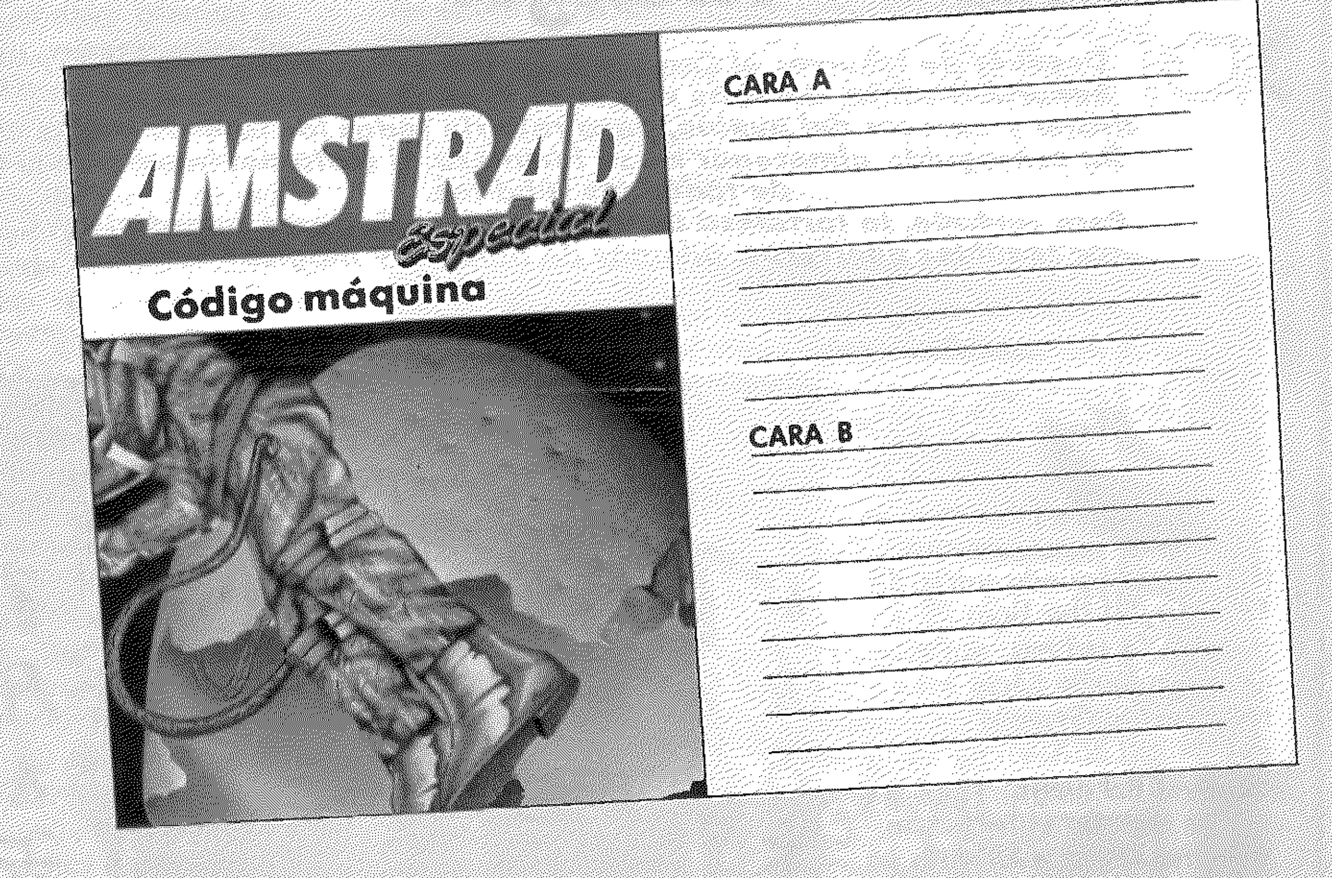


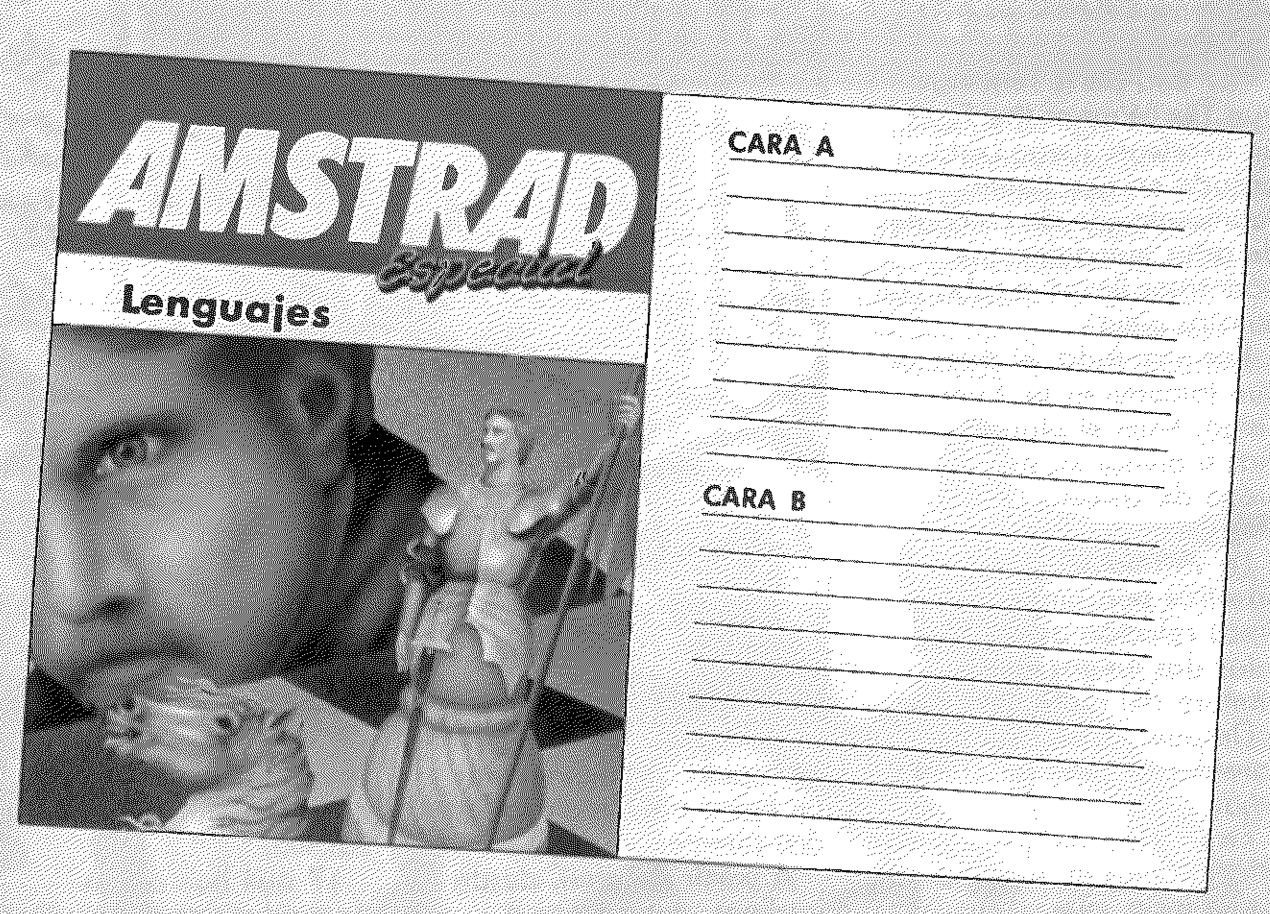
CARATULAS PARA SUS DISCOS

Para que guarde sus mejores programas de los, hasta ahora, cuatro especiales aparecidos, aquí tiene 4 carátulas para sus discos.






AMSTRAD
Especial
Código máquina
CARA A
CARA B


AMSTRAD
Especial
Lenguajes
CARA A
CARA B

Avenger

Después de recorrer el Camino del Tigre y lograr hacerte un honorable ninja, deberás vencer al Gran Guardián, principal tirano y opresor del pueblo, para robarle todos los tesoros que ha conseguido almacenar en su fortaleza, a costa del hambre de todos sus súbditos.
Pero cuidado, la fortaleza de tan terrible señor está guardada por temibles guerreros, además de odiosas y mágicas bestias.



Con Avenger se nos presenta la teórica segunda parte de The Way of the Tiger. Y decimos la teórica, porque no entendemos cómo los hombres de Gremlin, definen así el Avenger en la carátula del juego.

Nos extraña el hecho de que juegos tan diferentes en planteamiento se quieran relacionar.

Por una parte tenemos que The Way of the Tiger, un excelente juego, está basado en una buena representación de combates de artes marciales que podremos llevar a cabo con nuestro ninja.

Por otro lado, el Avenger, que también es un buen juego, tiene el planteamiento más o menos clásico del juego de «recolección» de objetos.

La verdad, es que nosotros pensamos que la única relación que une a estos dos juegos es el nombre del ninja, Avenger.

Después de estas consideraciones, pasemos a jugar.

En el inicio del juego, nos encontramos en los jardines del palacio, que, como todo buen jardín oriental, posee un tupido césped además de un típico arroyo cruzado por dos puentes.

A partir de aquí, desde los jardines, deberemos ir buscando las llaves que nos abran las puertas de las otras dependencias del castillo.

¿Y qué hay en las dependencias del castillo? Pues, evidentemente, los

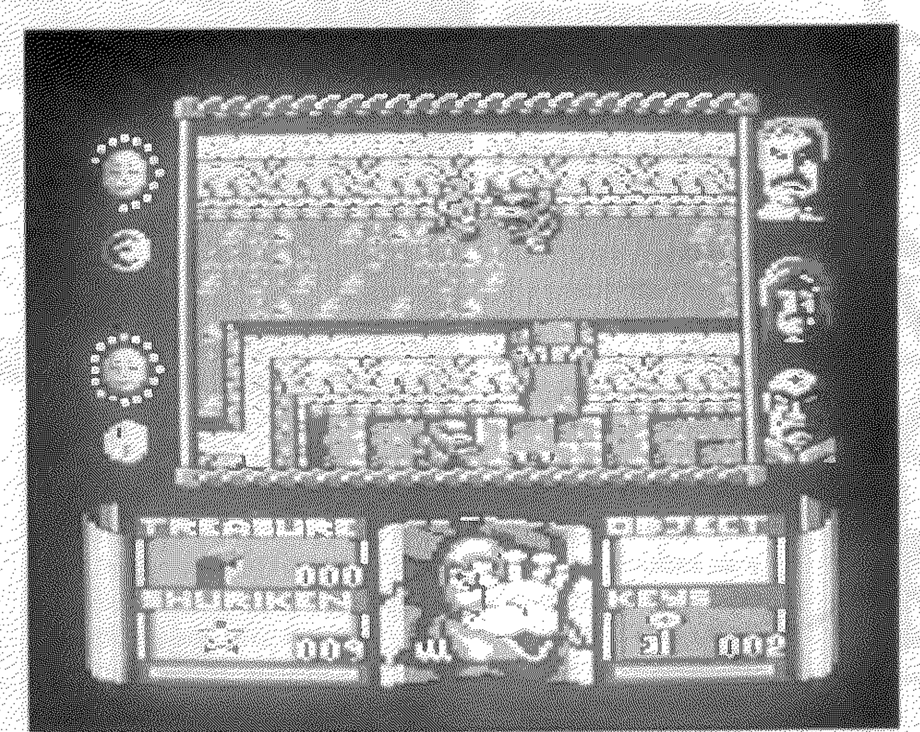
tesoros que tendremos que recoger para, como si de un Luis Candelas oriental se tratara, repartir entre los pobres.

Más, como a nadie le gusta que le roben el dinero, aunque a su vez él lo haya robado, ha colocado el amo y señor de palacio, todo tipo de fieros guerreros dispuestos a incordiar. Si a esto le añadimos algunas cobras y una especie de extraños insectos, que parecen mariquitas azules, tendremos un buen montón de motivos para no aburrirnos en tan encomiable misión.

Para ver el estado de salud en que se encuentra Avenger, disponemos en pantalla de unos curiosos indicadores.

Son dos esferas rodeadas de puntitos luminosos, 16 exactamente, que en la superior van desapareciendo con mayor o menor rapidez según la dureza del combate. Si el combate fuera muy duro, desaparecerían todos y en la esfera inferior perderíamos uno. Cuando en la esfera inferior sólo queden dos o tres puntos, será cuestión de ir planteándose la necesidad de pulsar la tecla perteneciente al número dos del teclado alfanumérico, si hacemos esto, nuestra energía se recargará y podremos seguir nuestra misión; si no, nuestras

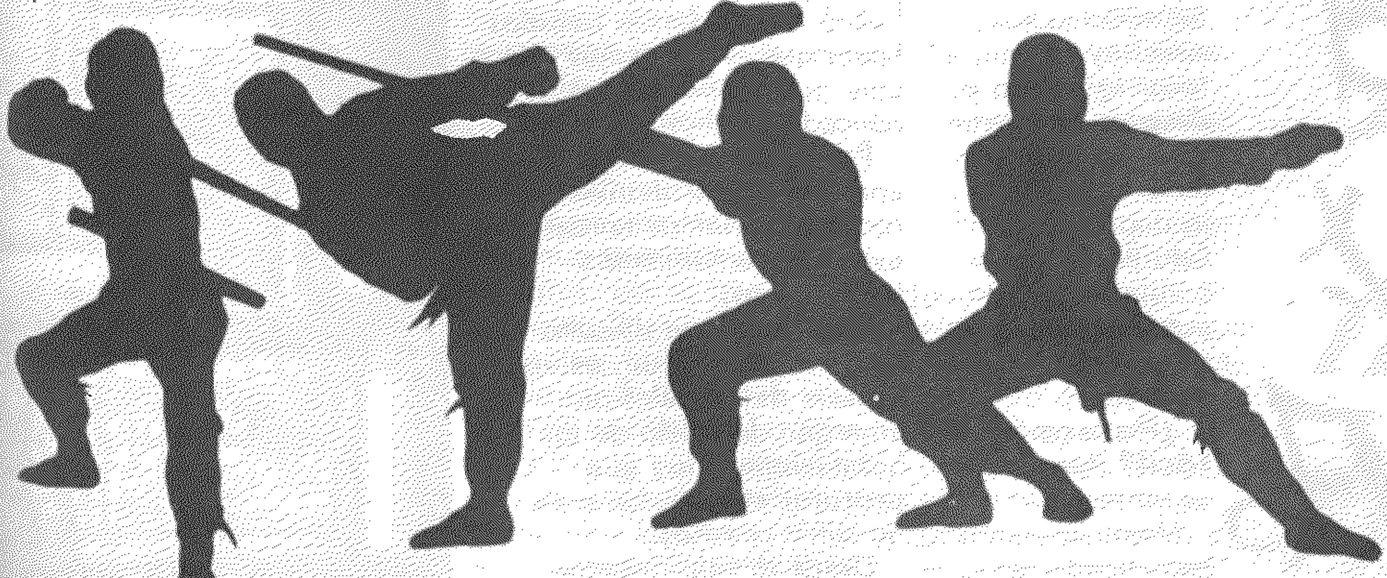


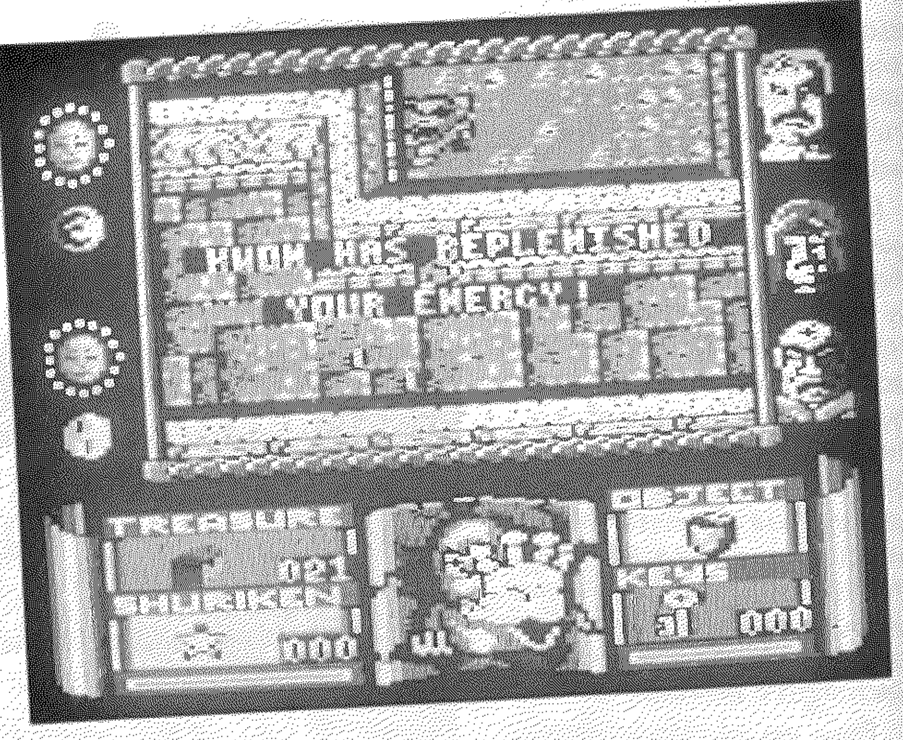


correrías están próximas a su fin.

Pero esta petición de energía vital a los cielos, sólo la podremos realizar en dos ocasiones. Si infringimos esto, nos saldrá en pantalla el mensaje de que los dioses se han enfurecido por nuestras llamadas y caeremos entre agónicos estertores muertos al suelo. Luego vendrá el eterno sufrimiento en el infierno.

Además de los indicadores





vitales, tenemos también otros que nos indican las llaves que llevamos, los tesoros que tenemos, los shuriken que nos quedan, y los objetos recogidos.

Una vez que sabemos cómo controlar nuestra situación, pasemos a la aventura.

Para no perder tiempo, deberemos ir hacia la parte inferior izquierda del jardín. Allí cogeremos la llave y el tesoro que veremos, sólo hay que tocarlos, y a toda velocidad, cosa nada difícil teniendo en cuenta que Avenger más que correr vuela, nos dirigiremos hacia la puerta de la dependencia que se halla en la zona más alta del jardín, laabriremos, teniendo la llave sólo hay que tocar la puerta y después de entrar recogeremos la llave que hay en la dependencia. Esta llave nos consigue automáticamente otras cinco en el marcador.

¿Parece fácil? Pues, no lo es tanto.

Mientras vamos recorriendo el jardín, deberemos enfrentarnos a guardianes con nunchakus, karatecas, extraños seres, bichos, etc.

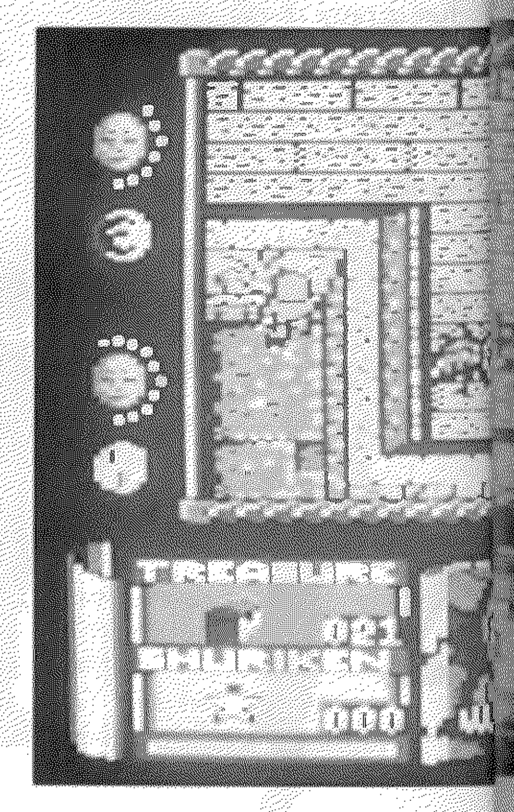
Y, aunque algunos son más difíciles de matar que otros, deberemos acabar con ellos a la máxima velocidad posible para que no nos resten mucha energía vital.

Un sistema rápido de hacer esto, es dispararles los shuriken con los que contamos que son 10. Cuando se nos gasten, habrá que ir dando patadas hasta que encontremos una caja que al tocarla nos concede 10 shuriken más.

Pero para no gastar ni energía vital ni shuriken, recomendaríamos también el evitar enfrentamientos innecesarios.

Otra cosa que dificulta nuestro movimiento, es una especie de cuchillos que emergen del suelo de las habitaciones, hay que procurar esquivarlos para no sufrir retrasos.

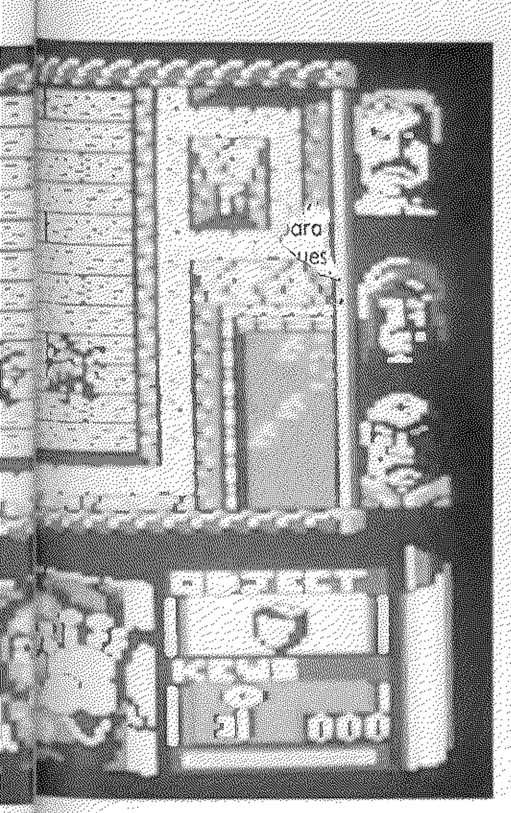
Desde la sala anterior deberemos dirigirnos a la que está inmediatamente debajo en la pantalla, allí recogeremos un tesoro, después abriremos una puerta y, girando a la izquierda, seguiremos por un corredor hasta el final. Allí abriremos una puerta y comenzaremos a reconocer las salas. Veremos una sala



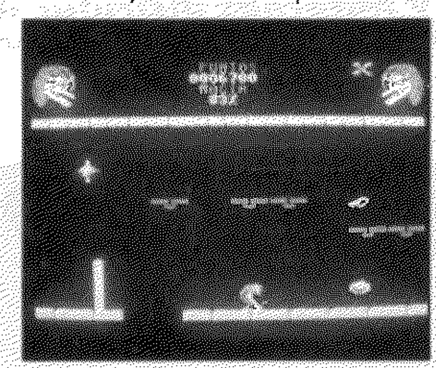


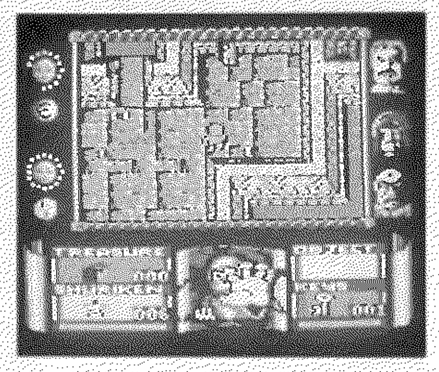
pequeña custodiada por un guerrero, nos le quitaremos de enmedio y recogeremos la llave que hay allí. Esto nos conseguirá tres llaves más.

Después, una vez que hayamos salido de nuevo al corredor, iremos hasta donde se cruza a la derecha con otro pasillo, pasamos a la primera sala de la derecha, batallamos con una «mariquita» y atravesamos una puerta, giramos a la izquierda y seguimos un corredor hasta que se cruza con otro, giramos a la izquierda, entramos por otro corredor *—¡vaya lío, eh!—*, lo seguimos hasta el final y



entramos en una sala de suelo de madera, cruzamos una puerta y nos peleamos bravamente con un guerrero, lo derrotamos y, ¡oh, maravilla!, vemos un objeto en el suelo, le recogemos con trémulas manos y un mensaje nos





dice que es un anillo que nos hará inmunes, eso sí, sólo por un momento. Locos de alegría, cruzamos la puerta que nos lleva a otra sala donde una necia mariquita nos ataca, gracias a nuestro anillo la vencemos sin despeinarnos y recogemos la preciada llave que está en el suelo.

Después..., bueno, la verdad es que no nos gustaría poneros las cosas excesivamente fáciles ni tampoco querríamos cansaros, por lo que pensamos que os dejaremos a vosotros la continuación.

Sólo os diremos que tendréis que bajar por una trampilla, la veréis en una sala de suelo de madera, y daros unas vueltas reconociendo la parte baja

Mister JOYSTICK

del palacio en donde tendréis que seguir cogiendo llaves y tesoros.

Bien, llegó la hora de las conclusiones y, la verdad sea dicha, éstas son positivas.

Sin que la trama sea nada espectacular es entretenida, consiguiendo el laberinto de corredores y habitaciones mantenernos abstraídos en la búsqueda de llaves y tesoros.

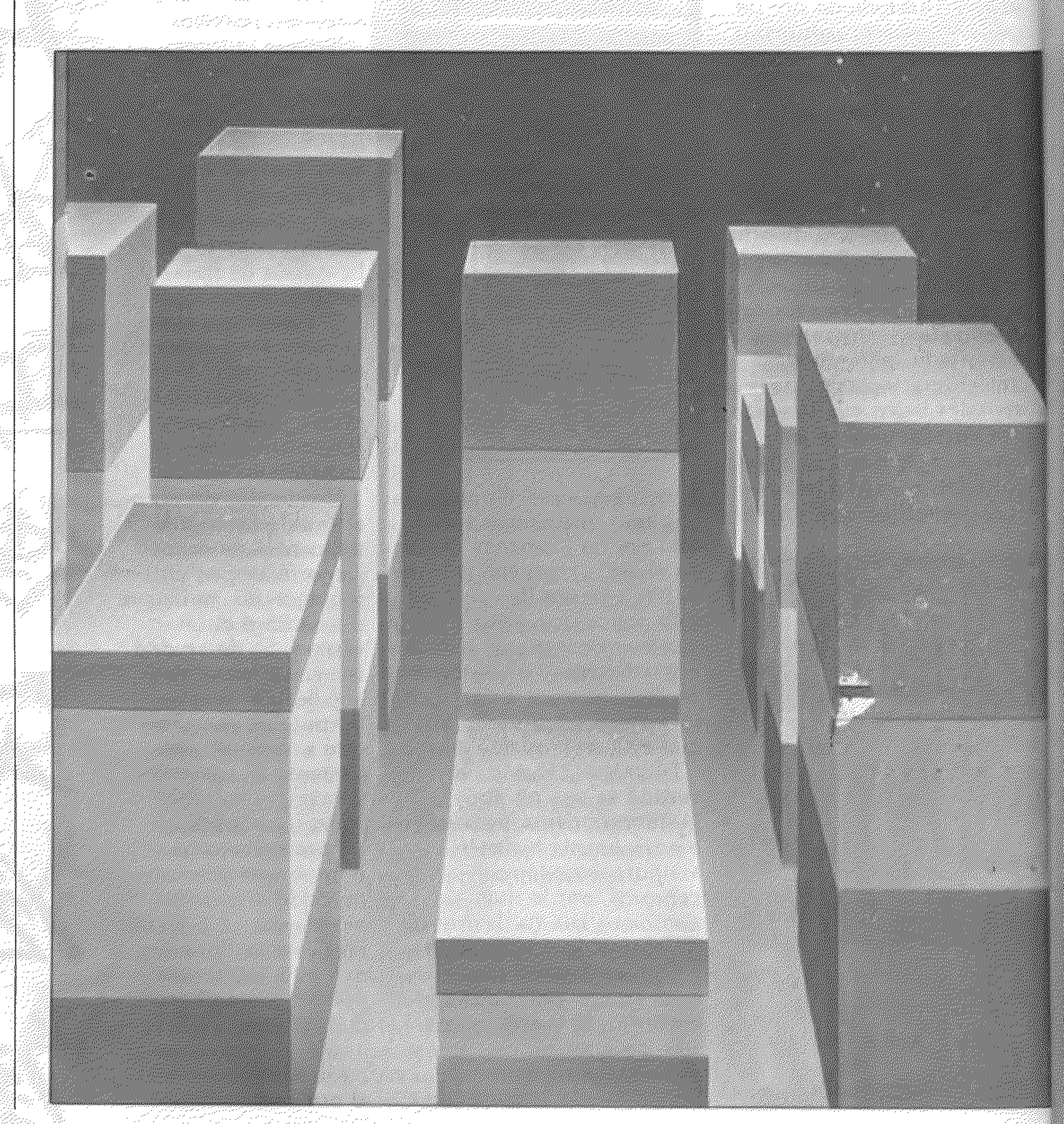
Los enemigos son bastante fáciles de eliminar, pero cuidado, no hay que olvidar que algunos quitan bastante energía.

Hemos dejado para el final lo realmente destacable de este juego. Avenger, el protagonista, se mueve a velocidad de vértigo. Sí, de verdad, se mueve como alma que lleva el diablo. Después de dirigir un rato a Avenger, se contagia su velocidad y vamos circulando como las **«motos»**.

Poco más que decir, si no es que nos ha parecido muy entretenido el Avenger de Gremlins Graphics, que está distribuido por Erbe y que, sin duda alguna, será un juego que nos haga pasar buenos ratos en el ordenador.

FORTH TE INVITA A JUGAR

Seguramente, bastantes de nuestros lectores, ya están un poco más convencidos de que el ordenador que tenemos instalado en casa** (no almacenado) **es algo más que un «juguete», destinado a ser utilizado para cargar en su memoria un buen programa de juegos y hacernos disfrutar de lo lindo con este simulacro de aventuras que ocurre en la pantalla.



Dentro de este grupo de personas se encuentran algunas *(cada vez más, por cierto)* que han comprobado por sí mismos que el lenguaje Basic no es el único utilizable para comunicarnos con nuestro querido **Amstrad**, sino que existe una gran variedad de formas y maneras de hacerlo. Estamos en lo cierto, **¿verdad?**

Este es el motivo que nos ha movido a continuar adentrándonos en el mundo de los lenguajes de programación ya que, además, nos parece que estamos ante una aplicación de nuestro ordenador casero que se sale de la clásica codificación de un programa en el lenguaje típico que casi todos ellos llevan incorporado: Basic.

El resultado de nuestros sondeos nos indicó que, sin miedo a equivocarnos, podíamos intentar ofrecer la posibilidad de poder trabajar con uno de ellos que, en principio por su forma de trabajo y por la propia concepción del lenguaje, nos resulta atractivo **«a tope»**. Y aquí es donde nos **«aliamos»** con el Forth.

Imaginamos que no habrá olvidado todavía el concepto de **«stack»** o **«Notación Polaca Invertida»**, por ejemplo, que les ofrecimos en el anterior número especial de esta revista. Vamos a estar utilizándolos casi continuamente, así que si todavía no tiene una idea clara de qué es cada una de estas cosas, para qué sirven y cómo se utilizan, le rogamos que eche un vistazo a todo lo que ya hemos publicado sobre ellas.

Como no podemos estarnos parados, le proponemos que esta vez avancemos un poco más en el tema, pero ahora que ya tenemos una cierta base teórica sobre el lenguaje y conocemos alguna de sus instrucciones o **«palabras»** ya definidas, lo haremos de una forma mucho más práctica, viendo que todas ellas pueden utilizarse para realizar **«cosas»** que sirvan para algo, pues su misión es bastante más amplia que repetir una cierta frase doce veces, por ejemplo. **¿O no?**

¡Manos a la obra! Lo primero será proyectar detenidamente **«qué»** es lo que queremos hacer, **¿de acuerdo?**

Nuestro objetivo va a ser muy sencillo. Intentaremos conseguir un programa para que el ordenador juegue con nosotros. No se tratará de **«algo»** difícil y complicado, sino de un mero entretenimiento que nos agudiza el ingenio y repetirá una vez más el enfrentamiento amistoso entre el **«hombre»** y la **«máquina»**. ¡Suerte!

Partimos de que encima de la mesa de juego hay 20 monedas, o cualquier otro número que nosotros elijamos. El primer jugador retirará de ella y a su criterio 1, 2, 3 ó 4 dejando el resto sobre la mesa.

El siguiente hará la misma operación con las que hayan quedado disponibles y el turno pasará, de nuevo, al jugador que inició la partida.

¿Hasta cuándo durará? La solución a esta pregunta es bien sencilla: si no hay monedas que retirar, se acabó lo que se daba. O más claramente diremos que el jugador que recoja la última moneda habrá ganado la partida dejando a su oponente a la altura del betún. *(Sugerimos darle la revancha para que después no haya quejas ni disculpas).*

Pero nuestro contrincante será duro de pelar ya que vamos a enfrentarnos con nuestro **Amstrad** y damos fe que posee una rapidez de reflejos digna de envidia. Así que, ¡cuidado!

¿Se atreve a realizarlo completamente en lenguaje Forth? Nosotros sí, de modo que le invitamos a acompañarnos en la empresa. ¿Entiende claramente **«qué»** pretendemos hacer? Seguro que sí.

Ahora nuestro problema será cómo hacerlo. La solución buscada podría conseguirse de la siguiente forma:

El programa tendrá que explicar primero en qué consiste el juego ya que aunque nosotros, sus autores, sabemos cómo funciona, un posible competidor seguro que no tiene ni idea de cómo empezar.

A continuación nos presentará en la pantalla la oportunidad de elegir quién comenzará a coger las monedas. Si no estamos dispuestos a dar el primer paso y le contestamos que no, seguro que el **Amstrad** está presto a empezar y realizará la primera jugada.

¿Ya sabemos quién comienza?, pues a pensar y calibrar para ver si logramos que no nos venza: entramos en la fase de «juego» propiamente dicho.

Supongamos que la competición se ha desarrollado sin ninguna incidencia ajena a las normas del juego y que la partida ya tiene ganador. **¿Qué hacer ahora?** El ordenador informará del resultado y vuelta a empezar.

Siguiendo este proceso no encontrará ninguna dificultad si desea plasmarlo gráficamente en un organigrama. Si no es así, observe la Figura I que le aclarará sus dudas.

COMIENZO
INFORMACION
SALIDA
JUEGO
FIN
FIN

FIGURA I

Todos sabemos o, por lo menos, debíamos, que el lenguaje Forth enriquece su vocabulario básico añadiéndole las **«palabras»** que nosotros mismos vamos definiendo basadas en las ya existentes. **¿Que por qué le decimos esto?** Vaya pensando en crear alguna nueva que cumpla con cada una de las funciones que hemos numerado anteriormente. Las llamaremos **«INFORMACION»**, **«SALIDA»**, **«JUGAR»** y **«FIN»** por similitud con cada uno de los procesos indicados en el organigrama.

Si le seguimos al pie de la letra traduciéndolo a **«palabras»** Forth conseguiremos definir la primera de las que está compuesto el programa. Prepare su intérprete Forth y ¡adelante! Teclee:

PROGRAMA I

```
: MONEDAS INFORMACION SALIDA JUEGO
FIN ;
```

y a continuación pulse la tecla RETURN.

¡Qué sorpresa! En lugar de encontrarnos con la definición de una nueva palabra, a la que hemos querido dar el nombre de **«MONEDAS»**, resulta que hemos obtenido un mensaje de error:

INFORMACION-Palabra no permitida

Claro, ¡cómo va a permitirnos el **«intérprete»** utilizar una **«palabra»** *INFORMACION)* que todavía no está definida y que, por tanto, no está disponible en el «diccionario»! Y podemos hacer extensivo esto a las demás: ninguna de ellas lo está, así que nuestro siguiente paso será decidir el modo de resolver cada uno de los gráficos de proceso *(rectángulos)* del organigrama que han dado lugar a estas **«palabras»**. No abandone el listado del Programa I ya que más tarde tendrá que volver a utilizarlo.

¿Qué es lo que tenemos pensado que haga la palabra **«INFORMACION»**? Tiene una misión realmente sencilla: contar al posible jugador las normas por las que se regirá la partida. En Basic sería muy sencillo fabricarnos una rutina que a base de instrucciones PRINT nos explique el juego. **¿Y en Forth?** No se preocupe, también es muy simple. Si no se lo cree, eche un vistazo al Programa II.

PROGRAMA II

```
: INFORMACION 0 CLG 6 SPACES ." VAMOS
A JUGAR CON NUESTRO ORDENADOR. DISPO
NEMOS DE 20 MONEDAS Y CADA" CR 15
SPACES ." JUGADOR IRA COGIENDO 1, 2,
3 O 4 A SU ELECCION." CR CR 15 SPACES
." GANARA EL QUE CONSIGA LLEVARSE LA
ULTIMA MONEDA." CR ;
```

No les mentíamos, ¿verdad? No emplea ninguna «palabra» fuera de las ya existentes en el repertorio básico del intérprete que estamos utilizando, así que dispóngase a definir la palabra **«INFORMACION»** a base de todas las que ya están. En esta ocasión no encontrará ningún obstáculo que se lo impida.

¿Novedades? Aparentemente ninguna. Siguiendo el listado vemos que primero «mete» un cero en el «stack» y luego borra la pantalla de gráficos con la palabra **«CLG»** y la tinta que contenga la posición más alta del stack: en este caso un cero.

A continuación y empleando ." *(equivalente al PRINT para textos)*, visualiza las líneas del mensaje. La palabra encargada de saltar línea es CR y SPACES tomará el valor que se encuentre en ese momento en el tope del stack e imprimirá tantos espacios como le indique dicho valor. **¿Quiere ver los resultados producidos por INFORMACION?** Teclee entonces, simplemente:

INFORMACION

y al instante podremos comprobar si lo que aparece en la pantalla es lo que pensábamos tras analizar la definición de la palabra. **¿Cómo queda el stack tras ejecutarla?** Veamos despacito este punto.

Primero introducimos en él un cero que después extraemos en CLG. Después metemos un 6 que sacamos con SPACES, imprimimos un texto y saltamos de línea *(estas dos palabras no manipulan el stack)*. A continuación ponemos un 15 para imprimir una serie de espacios en blanco, con lo que la pila vuelve a quedar vacía.

Si seguimos analizando la palabra INFORMACION veremos que al final el stack queda vacío ya que en él se introducen tantos valores como después se sacan. Compruébelo tecleando "." y le aparecerá un mensaje de error que le indica esta circunstancia. ¿Podemos dar como válida esta definición de palabra? En confianza, creemos que sí.

Continuemos. **«SALIDA»** tendrá que cumplir dos funciones: preguntarnos si queremos ser los primeros en jugar o no y guardar en el stack un indicativo de uno u otro jugador. Hemos decidido, por decreto, que después de ejecutarse esta palabra el elemento superior de la pila contenga un 1 si somos nosotros los que comenzamos el juego y un 0 en el caso que sea el ordenador. Su diagrama de flujo se corresponderá con el de la Figura II. **¿Intenta codificarla en Forth?** Ande, vaya al Programa III y vea cómo lo hemos hecho nosotros.

Empleamos una nueva «palabra» del repertorio básico: **«KEY»**, que mete en el tope del stack el código Ascii del carácter que se corresponda con la tecla que pulsemos.

PROGRAMA III

```
: SALIDA CR 25 SPACES ." SALES TU PRI
MERO? (S/N)" CR KEY DUP 83 = SWAP 11
5 = OR IF 1 ELSE 0 THEN ;
```

Luego duplicamos este valor ya que son dos las comparaciones que vamos a realizar (DUP). Veamos si es igual a 83 *(carácter 5)* por medio de:

83=

que deja en el stack un valor de «verdadero» o «falso» según haya ido la comparación a la vez que extrae del mismo los dos valores que hemos comparado.

Situación actual del stack tras la valoración de la condición: valor de la tecla en la zona baja y justo encima el resultado booleano de la comparación. Luego si queremos comprobar la posibilidad de que la tecla pulsada sea la «s» tendremos que intercambiar estos dos valores superiores para volver a colocar de nuevo el código de la tecla pulsada en el elemento superior. Po eso es por lo que empleamos la palabra SWAP y

115=

ya que 115 es el código Ascii de la letra «s».

Esta nueva comparación dejará otro valor booleano en el tope de la pila. Cuando al menos uno de ellos sea **«verdadero»**, lo que implica que habremos pulsado la **«s»** o la **«S»**, el resultado obtenido por la operación lógica OR será también verdadero si el IF que le sigue lo evaluara, eliminara los dos booleanos de la pila y colocara en su lugar un 1 tal como queríamos. En caso contrario *(pulsamos otra letra distinta)* el valor almacenado en el stack será un 0.

Resultado de la ejecución de la palabra **«SALIDA»**: Analiza la tecla pulsada, cuando sea **«S»** o **«s»** mete un 1 en el tope de la pila y un 0 en caso de no ser ninguna de las dos. En este momento sólo tendremos un valor almacenado en el stack, que depende de nuestra elección y que será el indicativo que utilizarán otras rutinas para saber quién ha salido. Compruébelo tecleando:

INFORMACION SALIDA

Bueno, ya tenemos resueltas dos de las cajas que forman el primitivo organigrama de la Figura I. Continuemos con las que nos quedan. La siguiente la hemos bautizado con el nombre **«JUEGO»**. Las funciones que tenemos previstas para él son las siguientes:

— Establecer el número de monedas con las que vamos a comenzar el juego *(20 en nuestro caso)*.

— Repetir por turno la jugada mientras queden monedas encima de la mesa.

— Eliminar el número de monedas almacenado cuando sea cero.

Y nada más, es así de sencillo. Su codificación será la del Programa IV en el que tras introducir en el stack un nuevo 20 *(valor inicial de la cantidad de monedas)* repite una llamada a la palabra **«TURNO»**, que ya definiremos, hasta que desaparezcan todas.

¡Ojo!, el intérprete tampoco nos aceptará esta nueva palabra hasta que no definamos **«TURNO»**, que se emplea repetidas veces en el bucle. El porqué ya está explicado anteriormente.

El paso siguiente será desarrollar la selección del turno de jugada para ver cómo distinguir a quién le toca coger las monedas y las acciones a realizar en cada caso. Su organigrama es el reflejado en la Figura III.

PROGRAMA IV

```
: JUEGO 20 DUP BEGIN 0 > WHILE CR
TURNO DUP REPEAT DROP ;
```

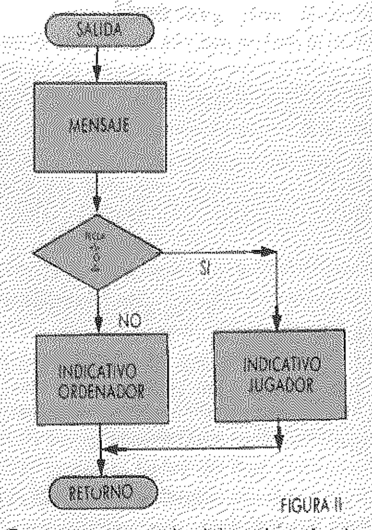


FIGURA II

Toma como partida del análisis el valor que la palabra **«SALIDA»** ha dejado en el stack según la tecla pulsada *(0 si sale el ordenador y 1 si comienza el otro jugador)* y podíamos descomponerlo de la siguiente forma:

— Analizar el valor que dentro del stack representa el indicativo del jugador.

— Si se trata del ordenador, invocamos a la palabra que ejecutará todo el proceso necesario para que elija su jugada y actúe en consecuencia.

— En caso contrario el que elije es el jugador.

— A continuación calculará las monedas que quedan en la mesa.

— Por último, cambia el indicativo del jugador actual por el del próximo (cambia el turno).

¿Pasamos a codificarlos? El programa V es la respuesta.

Al ver el listado pueden surgirnos varias preguntas. ¿A qué viene el **«OVER»** existente nada más comenzar? Intente ver cómo llega el stack al principio de la ejecución de esta **«palabra»**. En el tope contiene el número de monedas que quedan y en el elemento inmediato inferior el indicativo de cada jugador.

PROGRAMA V

```
: TURNO OVER 0= IF ORDENADOR ELSE
JUGADOR THEN QUEDAN SWAP 0= IF 1 ELSE
0 THEN SWAP ;
```

Si queremos analizar quién tiene en este momento el turno, es evidente que, copiando el

elemento que está debajo del superior en la pila en el nuevo tope de la misma, podremos hacer todas las comprobaciones necesarias ya que, de esta manera, tendremos disponible el número que representa a cada jugador.

Al final de esta palabra **«TURNO»** cambiamos la posición del stack que contiene el distintivo del jugador por el de su contrario para que vayan realizando cada uno su jugada alternativamente.

¿Cómo queda la pila después de esta rutina? En el elemento superior hemos dejado la cantidad de monedas que restan para que en la siguiente vuelta del bucle se compruebe si hemos llegado al fin o no. En el que está inmediatamente debajo guardamos el código del siguiente jugador, **¿de acuerdo?**

En esta **«palabra»** utilizamos otras que todavía no tenemos definidas. ¿Recuerda qué ocurrirá si la tecleamos así, directamente? Seguro que todavía sí.

La primera es **«ORDENADOR»** que se encargará de realizar la jugada del **Amstrad** de la siguiente manera:

— Si el número de monedas que restan es mayor que 4 elegirá un número aleatorio comprendido entre 0 y 4 que será las que retire de la mesa. Vaya pensando en una nueva «palabra» que lo haga.

— Si es menor que 5, el ordenador restará todas las que queden y ganará.

— En ambos casos nos informará de lo que ha hecho. En Forth el Programa que realiza todas estas acciones sería algo semejante al VI que resulta ser muy sencillito y que lo único que hace es añadir al stack un nuevo elemento que contiene las monedas elegidas para retirar.

Pero ahora habrá que definir la «palabra» que consiga entregarnos en el stack un número

PROGRAMA VI

```
: ORDENADOR DUP 4 > IF ALEATORIO ELSE
DUP THEN ." ELIJO " DUP . CR ;
```

aleatorio comprendido entre 1 y 4. Para ello, ahí está el Programa VII.

¿Qué hace? Nada extraño. Calcula un número aleatorio comprendido entre 0 y

PROGRAMA VII

```
: ALEATORIO RND 8192 / 1 + ;
```

32767, lo divide entre 8192 y le suma 1. **¿Le extraña el 8192?** ¡No se asuste, hombre! Divida 32767 entre 4 y obtendrá algo parecido. Lo hacemos de esta forma ya que aunque realmente lo suyo será calcular:

RND 4 * 32737 /

de esta forma podría producírsenos un **«overflow»** en uno de los elementos del stack y darnos el consabido mensaje de stack lleno.

Otra de las nuevas palabras, **«JUGADOR»,** será la que nos indicará que es nuestro turno. Para ello:

— Preguntará el número de monedas que hemos quitado del total.

— Analizará si el valor tecleado está comprendido entre los márgenes permitidos.

— Nos devolverá en el elemento superior de la pila la cantidad elegida.

Podemos suponer que esto implica dos procesos diferentes, uno de **«PREGUNTA»** y otro de **«COMPROBACION»**, así que si lo codificamos quedará como el Programa VIII.

PROGRAMA VIII

```
: JUGADOR PREGUNTA COMPROBACION CR ;
```

Y como siempre que ocurre esto nos encontramos con nuevas palabras, será necesario descender a un nivel inferior e intentar definirlas a su vez.

«PREGUNTA» se encarga de anotar el código Ascii de la tecla pulsada y restarle 48 *(que corresponde al 0)* para obtener en el elemento superior del stack el número elegido. *(Programa IX).*

PROGRAMA IX

```
: PREGUNTA ." ES TU TURNO, FORASTERO.
ELIGE. " KEY 48 - ;
```

Seguidamente **«COMPROBACION»** analiza si este valor está dentro de lo establecido *(1 a 4)* y en caso de no ser así eliminarlo, mediante un DROP, y volver de nuevo a la carga preguntándonos por un nuevo dato. Observe en su listado *(Programa X)* una cosa muy curiosa: es una palabra «recursiva», es decir, que se invoca a sí misma.

PROGRAMA X

```
: COMPROBACION DUP 5 < IF DUP 0 > IF
." CORRECTO." ELSE ." INCORRECTO."
DROP CR PREGUNTA COMPROBACION THEN
THEN ;
```

Bueno, ya tenemos en el stack tres datos: el indicativo del jugador, el número total de monedas y la cantidad a retirar. Para calcular las que quedan empleamos la palabra **«QUEDAN»** que, sencillamente, resta los dos elementos superiores *(total y las que quitamos)* y el resultado lo guarda de nuevo en la pila dejándola ahora con dos elementos. *(Programa XI).*

PROGRAMA XI

```
: QUEDAN - DUP ." QUEDAN " . CR ;
```

Y con esto ya tenemos totalmente claras y definidas cada una de las palabras que forman parte e intervienen dentro de la primitiva **«JUEGO»**. Sólo nos queda indicar que, una vez hemos salido del bucle que la forma, el juego toca a su fin y ya no necesitamos el número de monedas que quedan *(es evidente que no hay ninguna)*. Por eso utilizamos un DROP al final de la palabra **«JUEGO»** para

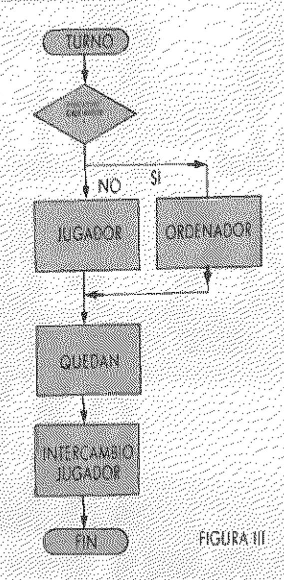


FIGURA III

eliminarlo quedándonos solamente con el distintivo del jugador que movería a continuación.

Y con este último elemento pasaríamos a ejecutar la palabra **«FIN»** en la que:

— Analizamos el siguiente jugador.

— Si es el ordenador *(o contiene un cero)*, quiere decir que hemos ganado nosotros y le concedemos la revancha.

— Si ocurre lo contrario, el **Amstrad** nos habrá dado una soberana paliza de la que es mejor no hablar.

PROGRAMA XII

```
: FIN 10 SPACES DUP 0= IF ." ME HAS "
ELSE ." TE HE " THEN ." GANADO. SI "
0= IF ." ME DAS " ELSE ." QUIERES "
THEN ." LA REVANCHA ESCRIBE MONEDAS."
CR ;
```

¿Alguna dificultad para comprender el listado del Programa XII? Recoge lo que tendría que hacer la palabra **«FIN».**

Ya está todo completo. Hemos descendido hasta el nivel más bajo de la definición de cada una de las nuevas palabras **«Forth»** comenzando desde lo más alto y general para terminar desarrollando tareas muy particulares y concretas que cumplan unos requisitos muy específicos.

Pero en este método existe la pega de no poder **«compilar»**, o **«interpretar»** en este caso, las palabras que contengan otras que todavía no estén definidas.

A pesar de ello le recomendamos que actúe así siempre que pueda, codificando sus programas de arriba a abajo sobre el papel y, una vez concluidos, pase a teclearlos en el **Amstrad** en el orden más conveniente para evitar sorpresas. **¿De acuerdo?**

Pues disfrute con este nuevo lenguaje Forth. Compruebe que el stack ha quedado vacío tecleando la palabra (.). De no ser así, algo no ha ido del todo bien según las normas.

Pues disfrute con este nuevo lenguaje Forth.

MICRO Manía

Año III · N 20 Sólo para adictos 350 Ptas.

Canarias, Ceuta y Melilla, 335 ptas.

¡No te lo pierdas!

Pokes para los 100 Mejores Juegos de Spectrum y Amstrad



FAIRLIGHT II

El mapa, los pokes y todas las claves de un juego «superdifícil»



GAUNTLET

Te descubrimos los secretos de un clásico de las máquinas recreativas

JUEGA CON NOSOTROS Y GANA UN EQUIPO DE MÚSICA

HOBBY PRESS

MONITOR DE DISCO

A ver, entre nosotros, ¿quién no está harto de que le sucedan cosas raras al disco, que normalmente acaban en la misteriosa desaparición de ficheros cruciales? Además, el disco es una especie de territorio vedado al sufrido usuario, algo así como «El Castillo Negro del Hechicero Negro». Ya es hora de remediar este estado de cosas, y «DiscEdit» es el programa de hacer malabarismos con el disco, el programa es corto, bonito y usa un sofisticado estilo de programar.

DiscEdit es un monitor de disco que le permitirá editar y manipular cualquier sector o pista. Siendo cuidadosos, incluso es factible recuperar ficheros borrados accidentalmente.

Para poder usarlo a pleno gas, hemos de tomar en consideración cómo trabaja el sistema de disco de **Amstrad**. Existen tres formatos estándar:

— Formato Sistema (CP/M).
— Formato IBM (IBM PC y CP/M compatible).
— Formato datos.

Todos ellos tienen algo en común: cada uno posee 40 pistas por disco, y los sectores son de una longitud de 512 bytes. Las principales diferencias son el número de sectores por pista y los números de sector.

Los distintos formatos de disco

Aquí hay una lista para cada formato:

— Formato sistema:
9 Sectores por pista
números de sector, 65 a 73
— Formato IBM:
8 sectores por pista
números de sector, 1 a 8
— Formato datos:
9 sectores por pista
números de sector, 193 a 201

No es necesario preocuparse del formato del disco que queramos editar, porque DiscEdit hará el trabajo por nosotros; lo anterior es sólo a título informativo, y para el que desee modificar el programa por su cuenta y riesgo, adaptándolo a sus necesidades concretas.

Funcionamiento de DiscEdit

Pasemos a estudiar cómo funciona DiscEdit.

Teclee el programa y sálvelo. Al arrancarlo, la pantalla cambiará a modo 2, se dibujará un espacio cerrado y se nos preguntará por una unidad de disco. Hay que responder 0 (para el disco A) o 1 (para el B).

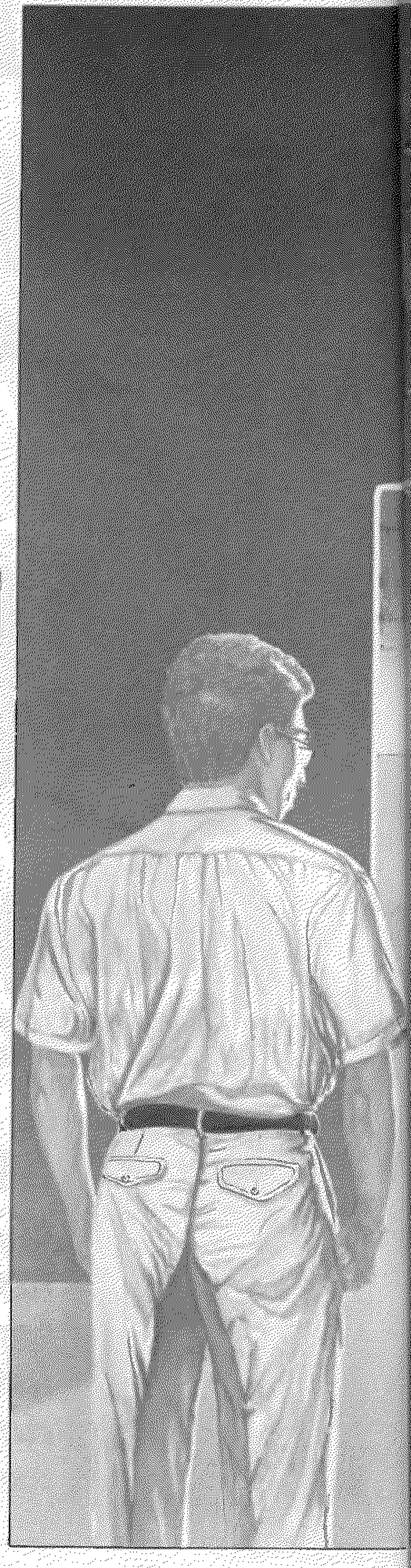
El disco seleccionado entrará en funciones durante un momento. DiscEdit hace esto para averiguar el formato y otras cosas; el resultado de la operación aparecerá en la esquina superior derecha de la pantalla.

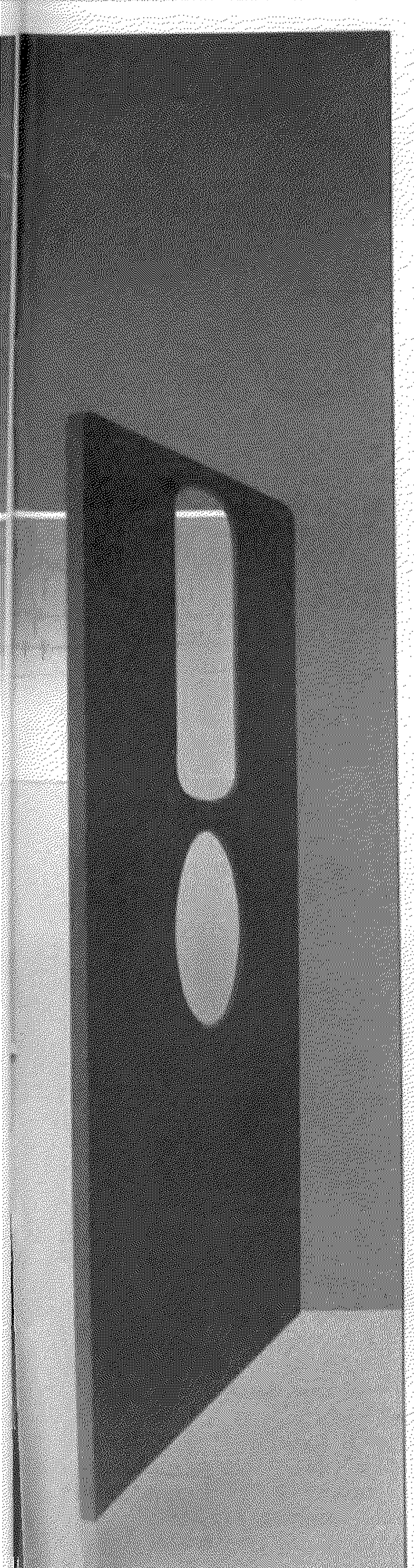
Acto seguido, hay que indicarle al programa la pista incial donde comenzará la edición (el **«track»**, un número comprendido entre 0 y 39. Lógicamente, después DiscEdit inquirirá acerca del sector que deseamos editar.

Como hemos indicado más arriba, el número de sector puede variar con el formato del disco; no se preocupe: el rango de valores aceptables aparecerá siempre en pantalla.

Al principio de cada línea hay un número hexadecimal que hace referencia al **«offset»** de los datos en el sector, esto es, el número del byte del sector a partir del cual vamos a editar. El resto de la línea muestra el contenido de ese byte, junto con el de los 15 bytes siguientes y su equivalencia en formato ASCII.

La pantalla no es lo suficientemente grande para mostrar los 512 bytes de un golpe, así que sólo aparecen





256. La otra mitad se obtiene pulsando la tecla **«2»** del teclado numérico (f2 en el 664 y 6128). Pulsándola una vez más, se retorna a la primera mitad. El número siguiente a «página» en la parte superior de la pantalla indica la porción que estamos viendo: 0 si es la primera ó 1 si es la segunda. A la izquierda del número de página están los de pista y sector en uso en ese momento.

Hay a su disposición una serie de opciones de edición como examinar, leer y escribir pistas y sectores. La tabla número 1 lista todo esto, junto con los controles de teclado correspondientes.

TABLA I

No obstante, la página de ayuda listará todos los comandos pulsando 0 del teclado numérico (fo en 664 y 6128).

Si ocurre algún error de disco en el proceso de edición, el mensaje aparecerá en la parte inferior de la pantalla junto con el consabido mensaje de **«RETRI, IGNORE O CANCEL?»**.

En este caso, es conveniente reintentar, porque a veces funciona; no tenemos ni idea de a qué se debe este comportamiento errático del **Amstrad.**

Cómo usar discEdit

Vamos a echar un vistazo a un típico ejemplo de cómo debiera usarse DiscEdit.

Primero de todo, el disco debe seleccionarse. En la mayoría de los casos, respondremos 0, señalando el disco A. Lo siguiente es escoger pista y sector: supongamos que introducimos 2 y 65, respectivamente.

El sector se encargará del disco y aparecerá en pantalla. Si, a pesar de todo, nos hemos equivocado, podemos desplazarnos a través del disco con las flechas del cursor, manteniendo apretada la tecla **«SHIFT»** (**«MAYS»** en el 6128).

Por ejemplo, pulsando MAYS+cursor derecha, se cargaría del disco el sector 66 de la pista 2. Una vez que estamos en el sector deseado, podemos comenzar a editar. Hay dos formas de hacerlo: en hexadecimal o en ASCII. Se selecciona uno y otro pulsando el 3 del teclado numérico (f3 para 664 y 6128).

En modo hexadecimal, los contenidos se alteran entrando dígitos del 0 al 9 y letras de la «a» a la «f». Observe que el teclado numérico no debe usarse para los números; use las teclas encima de «QUERTY».

El byte que está siendo editado puede cambiarse moviendo el cursor de edición al byte que queramos, pulsando las teclas del cursor apropiadas. La otra posibilidad de edición, como decíamos, permite entrar caracteres ASCII; por ejemplo, si se teclea ABCD, aparecerán los bytes &41, &42, &43 y &44.

Se puede saber en qué modo de edición estamos observando dónde se encuentra el cursor; si está encima de los números hexa, estamos en modo numérico, de lo contrario en ASCII. Una vez que el sector se ha dejado a nuestro gusto, podemos escribirlo en el disco. Para ello, basta pulsar la tecla 4 del teclado numérico (f4), y responder afirmativamente a la pregunta de si estamos seguros. Es el momento de descubrir cómo guarda el **Amstrad** la información en el disco.

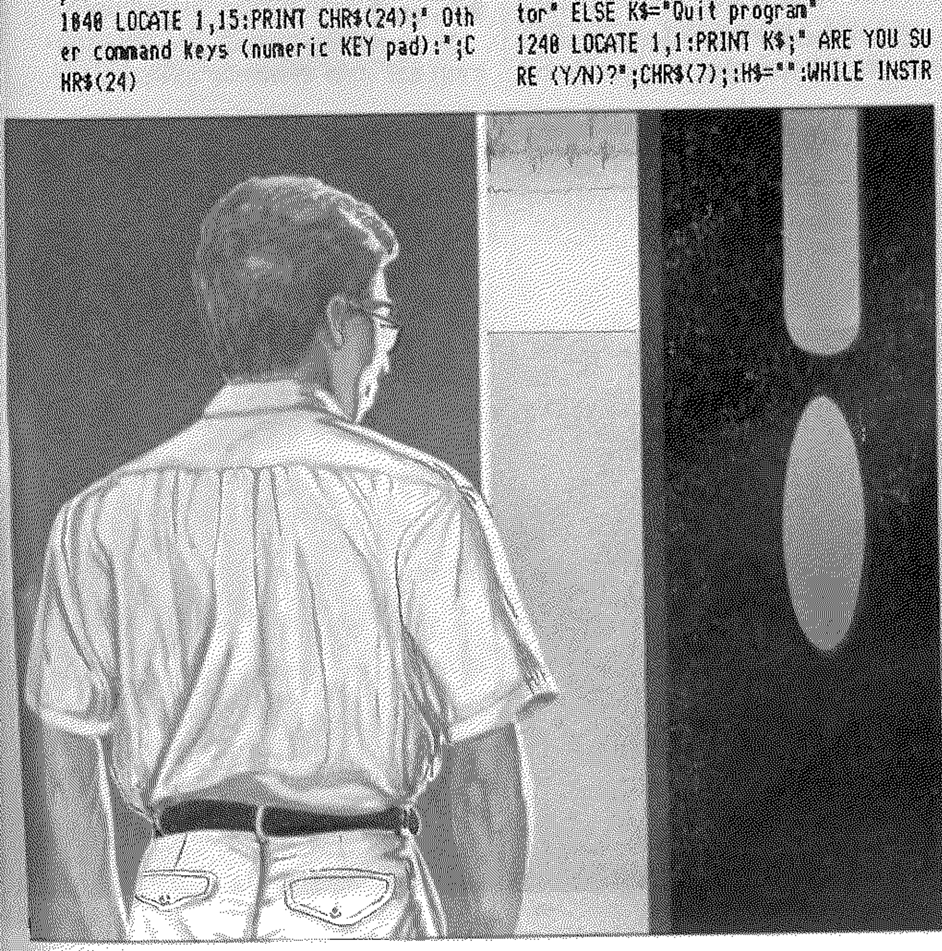
TABLA I

Opciones de pista/sector		
SHIFT	Cursor derecha	Siguiente sector
SHIFT	Cursor izquierda	Sector anterior
SHIFT	Cursor arriba	Avanza una pista
SHIFT	Cursor abajo	Retrocede una pista
Comandos de edición		
Cursor izquierda		un byte hacia la izquierda
Cursor derecha		un byte hacia la derecha
Cursor arriba		una línea arriba
Cursor abajo		una línea abajo
Otras teclas de control (Teclado numérico en el 464)		
0	(FO CPC664-6128)	Página de ayuda
1	(F1 CPC664-6128)	Nuevos parámetros de disco
2	(F2 CPC664-6128)	Cambio de página
3	(F3 CPC644-6128)	Modo de edición HEX/ACSII
4	(F4 CPC664-6128)	Escribir sector al disco
5	(F5 CPC644-6128)	Fin del programa

```
10 REM DISCEDIT
20 REM (C) AMSTRAD SEMANAL

30 REM
40 MODE 2:INK 0,1:INK 1,26:PAPER 0
50 MEMORY &7FFF:GOSUB 1270:' POKE MACHINE CODE
60 GOSUB 630:' INITIALISE
70 WINDOW SWAP 0,1
80 GOSUB 780:' DRAW SCREEN
90 CLS
100 T=0:CC=0:CR=0:WD=4:WI=24:OFF=64:V$=D$
110 PRINT CHR$(23);CHR$(1);:TAG #WD:GOSUB 540:POKE &908C,&80:GOTO 320
120 S$="":F=0:WHILE S$="":C=0:F=F XOR 1:MOVE OFF+CC*WI+T*8,334-CR*16:PRINT #WD,CHR$(143);:WHILE S$="" AND C<20:C=C+1:GOSUB 750:WEND:WEND:IF F=1 THEN MOVE OFF+CC*WI+T*8,334-CR*16:PRINT #WD,CHR$(143);
130 ADDRESS=PEEK(&908B)+PEEK(&908C)*256+CC+CR*16
140 IF ASC(S$)<237 THEN 350
150 IF ASC(S$)<239 THEN GOSUB 1230:IF W$="N" THEN 120
160 IF ASC(S$)=237 THEN K$="write":GOSUB 480:GOTO 120
170 IF ASC(S$)=238 THEN 1250
180 IF ASC(S$)<244 THEN 1140
190 IF ASC(S$)=246 THEN 1200
200 IF ASC(S$)=247 THEN GOSUB 920:GOSUB 780:GOSUB 820:GOTO 330
210 CC=0:CR=0:T=0:CLS
220 IF ASC(S$)=245 THEN POKE &908C,PEEK(&908C)XOR 1:GOTO 330
230 IF ASC(S$)=251 THEN sector=sector+1
240 IF ASC(S$)=250 THEN sector=sector-1
250 IF ASC(S$)=248 THEN track=track+1
260 IF ASC(S$)=249 THEN track=track-1
270 IF ASC(S$)=244 THEN GOSUB 540:POKE &908C,&80
280 IF sector>FSN+R THEN sector=FSN:track=track+1
290 IF sector<FSN THEN sector=FSN+R:track=track-1
300 IF track<0 THEN track=39
310 IF track>39 THEN track=0
320 K$="read":GOSUB 480:IF PEEK(&90E3)<>0 THEN 120
330 LOCATE #1,8,3:PRINT#1,drive:LOCATE #1,19,3:PRINT#1,track:LOCATE #1,31,3:PRINT#1,sector:LOCATE #1,42,3:PRINT #1,PEEK(&908C)AND &7F:LOCATE #1,58,3:PRINT#1,A$
340 LOCATE #2,1,1:WINDOW SWAP 0,2:TAG OFF #WD:CLS#3:CLS#4:CLS#5:CALL &9000:TAG #WD:WINDOW SWAP 0,2:GOTO 120
350 IF T=1 AND WD=4 THEN POKE ADDRESS,(PEEK(ADDRESS) AND &F0)+VAL("&"+S$)
360 IF T=0 AND WD=4 THEN POKE ADDRESS,(PEEK(ADDRESS) AND &F)+VAL("&"+S$)*16
370 IF WD=5 THEN POKE ADDRESS,ASC(S$)
380 LOCATE #1,9+CC*3,5+CR:K$=HEX$(PEEK(ADDRESS)):IF LEN(K$)=1 THEN PRINT#1,"0";
390 PRINT#1,K$:LOCATE #1,59+CC,5+CR:K$=".":IF PEEK(ADDRESS)>31 THEN K$=CHR$(PEEK(ADDRESS))
400 PRINT#1,K$;
410 IF WD=4 THEN T=T XOR 1:IF T=0 THEN CC=CC+1
420 IF WD=5 THEN CC=CC+1
430 IF CC=16 THEN CC=0:CR=CR+1
440 IF CC=-1 THEN CC=15:CR=CR-1
450 IF CR=16 THEN CR=0
460 IF CR=-1 THEN CR=15
470 GOTO 120
480 J=PEEK(&90E5):U=PEEK(&90E6)
490 POKE &90E5,track:POKE &90E4,drive:POKE &90E6,sector:POKE &90E3,0
500 IF ASC(K$)=114 THEN CALL &90B1 ELSE CALL &90C3
510 IF PEEK(&90E3)<>0 THEN CLS:PRINT"Disk ";K$;" error, Track:";track;"Sector:";sector;CHR$(7):track=J:sector=U:T$="":WHILE T$="":T$=INKEY$:WEND:CLS:RETURN
520 IF ASC(K$)=119 THEN PRINT"Sector written successfully";CHR$(7)
530 RETURN
540 GOSUB 820
550 CLS:LOCATE 1,1:PRINT"Input Drive: ";:nc=1:mi=0:ma=1:GOSUB 840:drive=VAL(w$):LOCATE #1,8,3:PRINT#1,drive
560 IF drive=0 THEN |A ELSE |B
570 FSN=PEEK(PEEK(DPH+16*DRIVE)+PEEK(DPH+1+16*DRIVE)*256+15)
580 R=8:IF FSN=&41 THEN a$="SYSTEM" ELSE IF FSN=&C1 THEN a$="DATA  " ELSE a$="IBM   ":R=7
590 LOCATE #1,58,3:PRINT#1,a$
600 CLS:LOCATE 1,1:PRINT"Input Track: ";:nc=2:mi=0:ma=39:GOSUB 840:track=VAL(w$):LOCATE #1,19,3:PRINT#1,track
610 CLS:LOCATE 1,1:PRINT CHR$(22);CHR$(1);"Input Sector (";CHR$(8);FSN;CHR$(8);"-";CHR$(8);FSN+R;CHR$(8);"): ";CHR$(22);CHR$(0);:nc=3:mi=FSN:ma=FSN+R:GOSUB 840:sector=VAL(w$):LOCATE #1,31,3:PRINT#1,sector;" "
620 CLS:RETURN
630 CALL &908E:'init machine code
640 DPH=PEEK(&BE40)+PEEK(&BE41)*256+18:D$="0123456789ABCDEF"
650 KEY DEF 14,0,245:KEY DEF 5,0,246
660 KEY DEF 15,0,247:KEY DEF 20,0,237
670 KEY DEF 12,0,238:KEY DEF 13,0,244
680 KEY DEF 2,1,241,249,249
690 KEY DEF 0,1,240,248,248
700 KEY DEF 8,1,242,250,250
710 KEY DEF 1,1,243,251,251
720 POKE &908B,0:POKE &908C,&80
730 WINDOW #1,1,80,22,25:WINDOW #2,2,80,5,25:WINDOW #3,2,6,5,20:WINDOW #4,9,56,5,20:WINDOW #5,59,75,5,20
740 RETURN
750 S$=INKEY$:IF (S$<CHR$(237) OR S$>CHR$(252)) AND INSTR(""+V$,UPPER$(S$))<2 AND LEN(V$)>0 THEN S$=""
760 IF V$<>"" THEN S$=UPPER$(S$)
770 RETURN
780 CLS #1:PEN #1,1:LOCATE #1,1,1:PRINT#1,"Amstrad Disc Sector Editor."
790 MOVE 0,380:DRAWR 216,0
800 MOVE 0,72:DRAWR 0,272:DRAWR 600,0:DRAWR 0,-272:DRAWR -600,0:MOVER 48,0:DRAWR 0,272:MOVER 405,0:DRAWR 0,-272
810 RETURN
820 LOCATE #1,1,3:PRINT#1,"Drive:      Track:      Sector:        page:      Format:        ":RETURN
830 nc=2:mi=2:ma=10
840 w$=""
850 s$="":WHILE s$="" OR LEN(w$)=nc AND s$<>CHR$(127) AND s$<>CHR$(13) OR s$=CHR$(127) AND w$="":s$=INKEY$:WEND
860 IF s$=CHR$(13) THEN 900
870 IF s$=CHR$(127) THEN w$=LEFT$(w$,
```

```
LEN(w$)-1):PRINT CHR$(8);CHR$(16);
880 IF s$<>CHR$(127) THEN PRINT s$;:w
$=w$+s$
890 GOTO 850
900 IF VAL(w$)<mi OR VAL(w$)>ma THEN
FOR a=1 TO LEN(w$):PRINT CHR$(8);CHR$
(16);:NEXT:PRINT CHR$(7);:GOTO 840
910 RETURN
920 WINDOW SWAP 0,1:CLS
930 LOCATE 31,1:PRINT CHR$(24);" *COM
MAND MENU* ";CHR$(24)
940 LOCATE 1,3:PRINT CHR$(24);" Movin
g to new track or sector:";CHR$(24)
950 PRINT"  <SHIFT> "+CHR$(243)+" ...
 move forward one sector."
960 PRINT"  <SHIFT> "+CHR$(242)+" ...
 move back one sector."
970 PRINT"  <SHIFT> "+CHR$(240)+" ...
 move forward one track."
980 PRINT"  <SHIFT> "+CHR$(241)+" ...
 move back one track."
990 LOCATE 1,9:PRINT CHR$(24);" Editi
ng commands:";CHR$(24)
1000 PRINT"  "+CHR$(243)+" ... move r
ight one byte."
1010 PRINT"  "+CHR$(242)+" ... move l
eft one byte."
1020 PRINT"  "+CHR$(241)+" ... move d
own one line."
1030 PRINT"  "+CHR$(240)+" ... move u
p one line."
1040 LOCATE 1,15:PRINT CHR$(24);" Oth
er command keys (numeric KEY pad):";C
HR$(24)
1050 PRINT"  f0 ... Help."
1060 PRINT"  f1 ... Select new disk p
arameters."
1070 PRINT"  f2 ... Page toggle, (tog
gles display between bytes"
1080 PRINT"                        0-2
55 & 256-511 of a sector)."
1090 PRINT"  f3 ... Toggles between H
EX & ASCII for editing."
1100 PRINT"  f4 ... Writes sector to
disk."
1110 PRINT"  f5 ... Exit program."
1120 LOCATE 30,24:PRINT CHR$(24);" PR
ESS SPACE TO EXIT ";CHR$(24):s$="":WH
ILE s$<>" ":s$=INKEY$:WEND
1130 CLS:WINDOW SWAP 0,1:RETURN
1140 IF ASC(S$)=243 THEN 410
1150 IF ASC(S$)=242 AND WD=4 THEN T=T
 XOR 1:IF T=1 THEN CC=CC-1
1160 IF ASC(S$)=242 AND WD=4 THEN 430
1170 IF ASC(S$)=242 AND WD=5 THEN CC=
CC-1:GOTO 430
1180 IF ASC(S$)=240 THEN CR=CR-1:GOTO
 450
1190 CR=CR+1:GOTO 450
1200 TAGOFF #WD:WD=WD XOR 1:TAG #WD:T
=0
1210 IF WD=4 THEN WI=24:OFF=64:V$=D$
ELSE WI=8:OFF=464:V$=""
1220 GOTO 120
1230 IF ASC(S$)=237 THEN K$="Save sec
tor" ELSE K$="Quit program"
1240 LOCATE 1,1:PRINT K$;" ARE YOU SU
RE (Y/N)?";CHR$(7);:H$="":WHILE INSTR
("xYN",H$)<2:H$=UPPER$(INKEY$):WEND:C
LS:RETURN
1250 TAGOFF #WD:PRINT CHR$(23);CHR$(0
):MODE 1:CALL &BB00
1260 END
1270 IF PEEK(&9000)=&DD THEN RETURN
1280 RESTORE 1360
1290 CHECKSUM=0
1300 FOR G=&9000 TO &90DC
1310 READ B$:POKE G,VAL("&"+B$)
1320 CHECKSUM=CHECKSUM+VAL("&"+B$)
1330 NEXT
1340 IF CHECKSUM=30232 THEN RETURN
1350 PRINT"M/C DATA ERROR";CHR$(7)
1360 DATA DD,2A,8B,90,DD,E5,C1,78,E6,
7F
1370 DATA 32,8D,90,CD,62,90,79,32,8D,
90
1380 DATA CD,62,90,DD,E5,06,03,CD,81,
90
1390 DATA 06,10,DD,7E,00,32,8D,90,CD,
62
1400 DATA 90,3E,20,CD,5A,BB,DD,23,10,
EE
1410 DATA 06,02,CD,81,90,DD,E1,06,10,
DD
1420 DATA 7E,00,FE,20,D2,45,90,3E,2E,
C5
1430 DATA CD,5D,BB,C1,DD,23,10,ED,3E,
0D
1440 DATA CD,5A,BB,3E,0A,CD,5A,BB,DD,
E5
1450 DATA C1,3E,00,B9,C2,04,90,C9,21,
8D
1460 DATA 90,ED,6F,CD,70,90,ED,6F,CD,
70
1470 DATA 90,C9,E6,0F,C6,30,FE,3A,FA,
7B
1480 DATA 90,C6,07,C5,CD,5A,BB,C1,C9,
C5
1490 DATA 3E,09,CD,5A,BB,C1,10,F7,C9,
00
1500 DATA 00,00,DD,21,DD,90,21,DB,90,
CD
1510 DATA D4,BC,DD,75,00,DD,74,01,DD,
71
1520 DATA 02,21,DC,90,CD,D4,BC,DD,75,
03
1530 DATA DD,74,04,DD,71,05,C9,21,00,
80
1540 DATA ED,5B,E4,90,3A,E6,90,4F,DF,
DD
1550 DATA 90,D2,D5,90,C9,21,00,80,ED,
5B
1560 DATA E4,90,3A,E6,90,4F,DF,E0,90,
D2
1570 DATA D5,90,C9,3E,FF,32,E3,90,C9,
84
1580 DATA 85
```



¿POR QUE NO TRUCAR EL CP/M 3.0?

Francisco G. R.

De todos es sabido los trucos que podemos utilizar en Basic para nuestras aplicaciones, y tan sigilosamente guardamos para nuestros programas. Pues bien, éstas también las podemos encontrar en el CP/M.

¿Cómo crearnos un fichero autoejecutable desde CP/M?

Lo podemos hacer de varias maneras, una de ella sería utilizando un procesador de textos e introducir los mandatos u órdenes deseadas. Por ejemplo, supongamos que cuando cargamos el CP/M, necesariamente tenemos que preparar el canal de la impresora y a continuación ejecutar un programa:

DEVICE CONIN=LPT
PROGRAMA. COM

Esto nos dará una entrada al canal de impresora para mandar toda la salida del monitor a la impresora. Si no dispusiéramos de un procesador de textos los haríamos desde Basic de la siguiente forma:

```
10 OPENOUT "DISC.SUB"
20 PRINT #9, "DEVICE CONIN=LPT"
30 PRINT #9, "PROGRAMA. COM"
40 CLOSEOUT
```

Lo último que nos falta decir es el nombre del fichero y cómo ejecutarlo. Si lo queremos ejecutar, el nombre del fichero cargador sería "PROFILE. SUB", si lo queremos cargar cuando entremos en el sistema sería un título diferente al de "PROFILE". Pero además debe de estar presente en el disco del programa "SUBMIT.COM", que es el encargado de leer los ficheros terminados en 'SUB.

¿Cómo simular un arranque tipo profesional de IBM, en el cual nos pregunte la fecha y la hora?

La creación del fichero autoejecutable está descrita anteriormente, ahora tan sólo tendríamos que introducirle los ficheros adecuados para que pueda ejecutar esta orden. El fichero encargado de cargar el reloj del equipo y ponerlo en hora se denomina 'DATE'. Si el primer fichero que introducimos es el de 'DATE', nos mostrará la fecha que tiene implícita y el día, también nos dirá el tiempo que llevamos con el ordenador encendido *(o bien desde la última vez que efectuamos un Reset)*. Si le damos la opción de:

DATE CONTINUOUS *(también es válido sólo la C)*

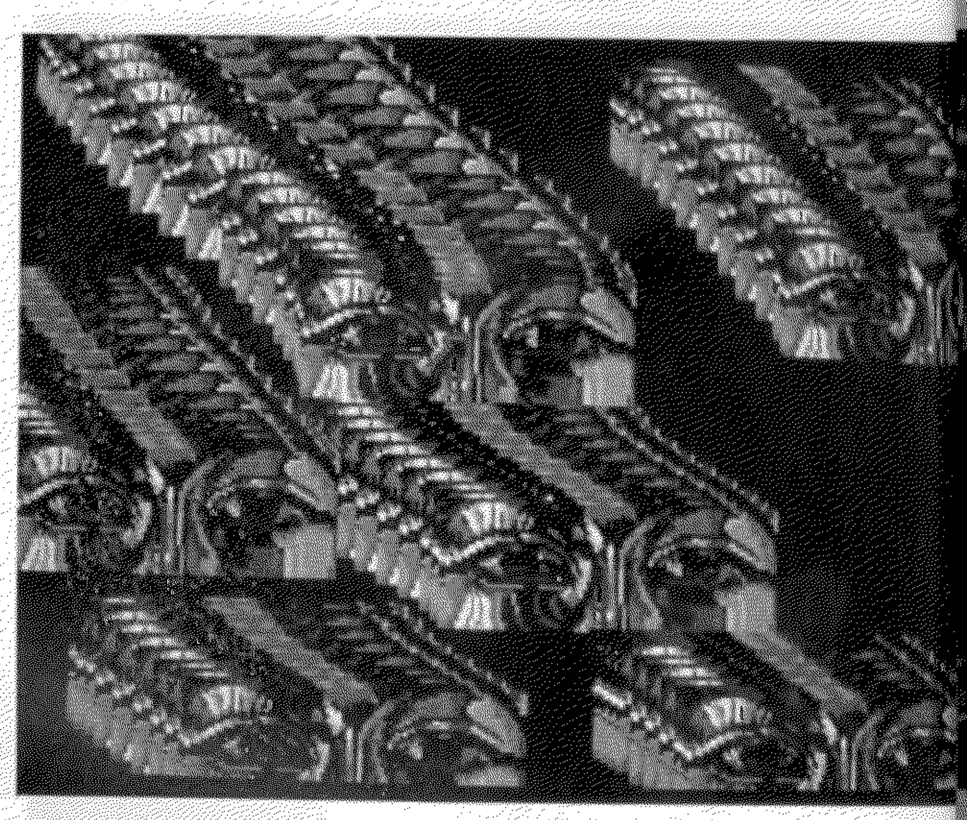
El ordenador nos mostrará el reloj en tiempo real que podemos ver cómo va avanzando segundo a segundo. Pero si queremos tan sólo que nos pida la fecha y la hora la opción será la siguiente:

DATE SET

El ordenador nos pedirá primero la fecha y a continuación la hora, si bien pulsando la tecla de Return dejaríamos el valor que tiene por defecto.

Todos sabemos cómo sacar un listado desde CP/M con el comando TYPE PROGRAMA.TXT, pero si el fichero tiene por extensión un COM, REL, etc., el ordenador nos muestra una serie de símbolos y caracteres no legibles, **¿podríamos listar un programa con la extensión COM, REL, etc.?**

Sí, es posible ¿pero cómo?, pues bien tan sólo debemos utilizar un



la capacidad de los programas que tiene en el disco. Usted podría hacer dos cosas, la primera era irse a Basic, efectuar un CAT, en el cual sí le dice la capacidad, y la segunda sería utilizar un DIR [FULL]. Pero esta última opción es un poco enmarañada, y nosotros le vamos a dar una en la que se sienta tan familiar como el CAT de Basic:

DIR [SIZE]

También podemos excluir ficheros cuando deseemos sacar un directorio del disco, por ejemplo, cuando tenemos muchos programas en el disco y sólo queramos sacar algunos en determinado. Esto lo podemos conseguir con el atributo entre corchetes [EXCLUDE=PROGRAMA.EXT]. Hablando de excluir programas si sólo queremos sacar los programas que estén en 'READ ONLY', *(programas sólo de lectura)* el atributo será 'RO', y si sólo queremos los 'READ WRITE', *(caso contrario al anterior, escritura y lectura)* el atributo sería 'RW'. Para finalizar esta parte de excluir programas o sólo sacar programas en determinados hay otra opción que nos muestra los programas que son invisibles *(programas en sistema)*, que se denota como 'SYS', aunque también se puede efectuar con el comando DIRS o DIRSYS.

Para acabar las opciones DIR vamos con una muy curiosa, consiste en sacar un número de líneas determinado a partir del encabezamiento. O sea, que podemos listar los programas del disco de tres en tres líneas o de diez en diez programas *(cinco líneas)*. El comando es el siguiente:

DIR [LENGTH=10]

A lo mejor cuando estamos borrando ficheros desde CP/M, nos encontramos que borramos alguno que no queríamos, sobre todo cuando borramos una serie de éstos que comienzan con la misma letra, pero hay uno de ellos que no queríamos borrar y lo hemos destruido. Pues bien, nosotros podemos obligarle al ordenador que nos pida permiso o conformidad para que efectúe el borrado de cualquier programa. Esto lo realizaremos de la siguiente forma:

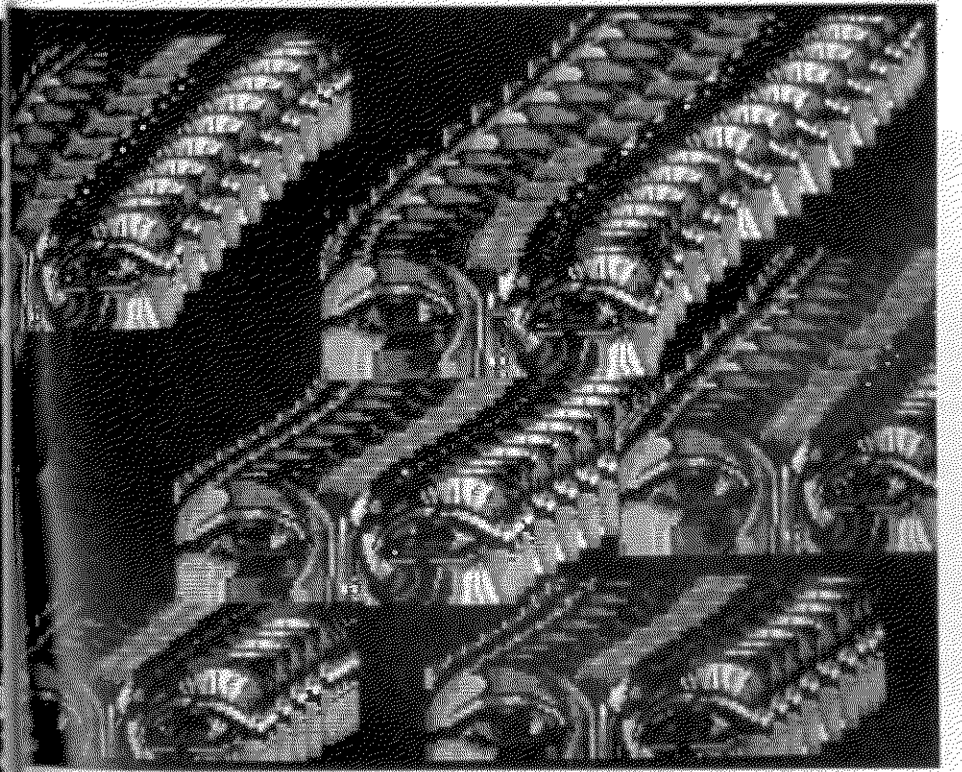
ERAse [CONFIRM] *(o bien tan sólo C)*

De esta forma tan sólo nos borrará los programas que nosotros le indiquemos.

comando de CP/M denominado 'DUMP'. El comando DUMP nos da un listado del programa en dos formatos: una a la izquierda con los códigos hexadecimales, y una segunda a la derecha en forma ASCII. De esta forma, y si previamente hemos instalado la impresora, podremos realizar un listado por impresora o bien por pantalla de los programas que no podemos acceder a ellos mediante un TYPE.

¿Cómo utiliza usted el comando DIR? ¿Le gustaría sacarle más jugo a este comando?

Si está dispuesto, adelante. Posiblemente usted no se sienta satisfecho con sólo utilizar el comando DIR a secas, pues muchas veces le gustaría conocer



IVA

DISTRIBUIDOR ZELIG SOFTWARE
AMSTRAD CPC 6128/664/464 CON DISCO

Víctor Prieto

Profesionales liberales, autónomos y demás personas acosadas por el IVA, hallarán en el Control del IVA de Zelig, la solución cómoda a los quebraderos de cabeza de las declaraciones trimestrales.

El ingreso de España en la Comunidad no sólo nos ha dado un carácter más europeo y ha propiciado la apertura de las fronteras al comercio, además nos ha traído un regalo que encontramos hasta en la sopa: el IVA.

Antes, con pagar simplemente bastaba, ahora hemos de pagar el recargo del IVA, ahora todos somos recaudadores de Hacienda y el IVA que se nos abona hemos de devolverlo escrupulosamente, pues como dice la canción: «Todos somos prisioneros de Hacienda».

Como con esta entidad las cuentas deben ser claras y Hacienda tarde o temprano acaba enterándose de todo, gracias a su potencial informático, ¿por qué no utilizar sus mismas armas?

Poniendo a la informática de nuestro lado, podemos llevar un control completo de nuestras facturas con el programa de Zelig.

Características

El Control de IVA, tiene capacidad para manejar hasta 150 cuentas entre clientes y proveedores, cada una de las cuales ha de ser identificada por un código de cinco dígitos, los cuales pueden ser ajustados a la normativa exigida en el Plan General Contable. En este caso, los clientes quedarán agrupados en el subgrupo 43, y los proveedores en los capítulos 40, 41, 42.

Por otra parte el fichero de apuntes que es capaz de soportar el programa, asciende a 2.000 anotaciones, susceptibles de ser borrados una vez alcanzado el tope.

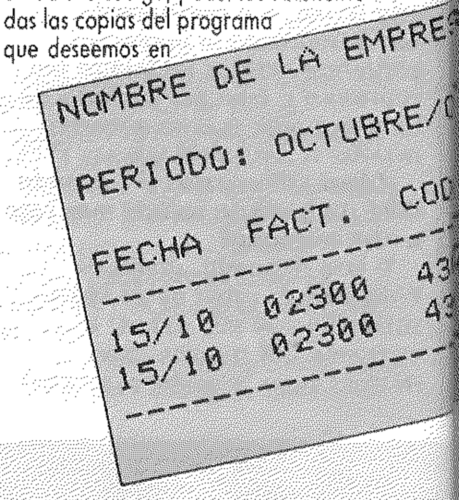
Con tal capacidad cualquier profesional o autónomo podrá llevar perfectamente su IVA sin ningún tipo de limitación, debido a que el número de cuentas y movimientos en un principio es suficiente para cualquier aplicación, aunque ésta se extienda a empresas pequeñas.

Personalización

Como todos los programas de la casa, éste puede ser personalizado, cosa que no resulta obligatoria, en nuestro caso si elegimos la personalización, nuestro nombre aparecerá en listados e informes impresos, en caso de no proceder a ésta, también puede usarse sin ningún tipo de problemas.

En este caso no existe código de cliente, con lo cual evitamos el tema de tener que introducirlo en la aplicación número 100 del mismo, evitando el riesgo de que por error u olvido éste quede totalmente inutilizado.

Por el mismo motivo, y ante la ausencia del antedicho código, podemos obtener todas las copias del programa que deseemos en



NOMBRE DE LA EMPRESA: INVESTIGACION SA — PAGINA 1

PERIODO: OCTUBRE/OCTUBRE — LIBRO REGISTRO DE FACTURAS RECIBIDAS

FECHA	RECEP	COD.	PROVEEDOR	B. IMPONIBLE	TIPO	CUOTA
05/10	01000	40001	PAPELERIAS FOLEO DE ORO	15,000	12%	1,800
05/10	01000	40001	PAPELERIAS FOLEO DE ORO	15,000	5%	750
12/10	01001	40002	PINUTRAS JACINTO SA.	40,000	12%	4,800
12/10	01001	40002	PINUTRAS JACINTO SA.	40,000	5%	2,000
15/10	01002	40003	MOBILIARIO GARCIA Y CIA.	200,000	30%	60,000
15/10	01002	40003	MOBILIARIO GARCIA Y CIA.	200,000	7%	14,000
12/10	01003	40004	DECORACION AMBIENTE SA.	75,000	12%	9,000
12/10	01003	40004	DECORACION AMBIENTE SA.	75,000	5%	3,750
07/10	01004	40005	MATERAL OFICINA PEREZ	34,000	12%	4,080
07/10	01004	40005	MATERAL OFICINA PEREZ	34,000	5%	1,700
22/10	01005	40006	INDUSTRIAS CORRAL E HIJOS	77,000	12%	9,240
22/10	01005	40006	INDUSTRIAS CORRAL E HIJOS	77,000	5%	3,850
15/10	01006	40007	COMPUESTOS ACIDOS SA.	23,000	12%	2,760
15/10	01006	40007	COMPUESTOS ACIDOS SA.	23,000	5%	1,150
			TOTAL I.V.A. SOPORTADO:			118,880

la completa confianza de que éstos van a funcionar sin ningún tipo de problema.

Una vez realizada la personalización podemos poner en marcha el programa y empezar a trabajar para Hacienda.

Menú principal

Con el siguiente repertorio funcional podemos elegir la actividad deseada.

F1: Introducción de apuntes.
F2: Consultas y listados
F3: Altas clientes/proveedores
F4: Bajas clientes/proveedores
F5: Listado clientes/proveedores
F6: Inicialización de ficheros
7: Corrección de apuntes

Para empezar a trabajar con el programa, la primera operación a realizar es la de dar de alta a los clientes y proveedores, operaciones que es aconsejable hacer utilizando la codificación ajustada al Plan General Contable.

Utilizando la opción n.° 3 del menú procedemos a rellenar el fichero de clientes y proveedores, cada uno de los cuales dispone de un código numérico de 5 cifras y una longitud de texto de 28 caracteres.

En caso de error, y siempre antes de pulsar la tecla ENTER, podemos volver al apartado anterior, es decir el código con un toque de tecla, en caso de advertir nuestro error después del ENTER, la opción F4 nos saca del apuro borrando de los ficheros el cliente y su código.

Diario de apuntes

Una vez dados de alta todos los profesionales y empresas con los que vamos a trabajar, podemos iniciar el proceso de anotaciones y apuntes en el diario.

Este está estructurado en forma de fichas, cada una de las cuales contiene los siguientes apartados:

Mes del apunte
Número de factura
Código de cliente/proveedor
Fecha de la factura
Base imponible
Tipo de IVA
Recargo

Observando el número de apartados y su índole podemos advertir que se contemplase todos los datos interesantes y de utilidad en cada factura, con los cuales llevar el control de nuestro IVA.

La inserción de datos en cada campo se realiza de forma cómoda y sólo es necesario introducir valores numéricos, el ordenador se encarga de buscar en memoria el artículo al que corresponde el código e imprimirlo en su correspondiente apartado. En este campo, al teclear el número de mes aparecerá automáticamente escrito el nombre de ésta, siguiendo similar proceso al poner el código de cliente aparecerá el nombre del mismo, una forma acertada de captar los datos. Como siempre en caso de error y antes de que la ficha quede grabada, podemos volver a cualquier campo anterior y modificar lo que deseemos.

Consultas e información impresa

Toda la información puede ser obtenida por pantalla o impresora, en la opción F2 que es la utilizada para estos menesteres podemos conseguir los siguientes datos:

Listado de facturas emitidas
Listado de facturas recibidas
Cuentas con clientes
Cuentas con proveedores

Las consultas de facturas se realizan por fecha, determinando el comienzo y fin de dicha consulta, para las cuentas de clientes o proveedores, ésta se realiza entre los plazos fijados y la entidad o persona elegida.

Fuera de esta opción podemos obtener un completo listado de todos los clientes y proveedores mediante la F5, en el cual aparecerán separados en dos grupos según la índole de cada uno.

Libro de apuntes llenos

En caso de agotar la capacidad de registrar apuntes del programa 2.000, podemos proceder al borrado de los mismos para continuar trabajando. Antes de realizar tal operación es imprescindible obtener un listado de todos los movimientos, con el cual guardaremos una información indeleble para cualquier consulta. Otro método válido, puede ser realizar una copia completa del disco; programa+ficheros,

```
A: INVESTIGACION SA                                  PAGINA  1
        COD: 43002 CLIENTE: FOTOGRAFIA JULIAN GOMEZ
TUBRE                     B. IMPONIBLE TIPO              CUOTA
   CLIENTE
.  FOTOGRAFIA JULIAN GOMEZ        50,000  12%            6,000
2  FOTOGRAFIA JULIAN GOMEZ        50,000   5%            2,500
              TOTAL I.V.A. REPERCURTIDO:                 8,500
```

NOMBRE DE LA EMPRESA: INVESTIGACION SA — PAGINA 1

PERIODO: OCTUBRE/OCTUBRE — LIBRO REGISTRO DE FACTURAS EMITIDAS

FECHA	FACT.	COD.	CLIENTE	B. IMPONIBLE	TIPO	CUOTA
12/10	04560	43001	INDSUTRIAS COMERCIAL	40,000	12%	4,800
12/10	04560	43001	INDSUTRIAS COMERCIAL	40,000	5%	2,000
15/10	02300	43002	FOTOGRAFIA JULIAN GOMEZ	50,000	12%	6,000
15/10	02300	43002	FOTOGRAFIA JULIAN GOMEZ	50,000	5%	2,500
20/10	43001	43003	ESTUDIO ALFA PUBLICIDAD	100,000	12%	12,000
20/10	43001	43003	ESTUDIO ALFA PUBLICIDAD	100,000	5%	5,000
17/10	01222	43004	AGENCIA INFORMACION & CO.	35,000	12%	4,200
17/10	01222	43004	AGENCIA INFORMACION & CO.	35,000	5%	1,750
08/10	43005	43005	IDIOMAS EL INGLES	70,000	12%	8,400
08/10	43005	43005	IDIOMAS EL INGLES	70,000	5%	3,500
19/10	04220	43006	ACADEMIA MATEMATICA Y CIA	23,000	12%	2,760
19/10	04220	43006	ACADEMIA MATEMATICA Y CIA	23,000	5%	1,150
19/10	55001	43007	CONDUCCIONES EL CANAL	120,000	12%	14,400
19/10	55001	43007	CONDUCCIONES EL CANAL	120,000	5%	6,000
14/10	23001	43008	ALCHOLERA GRANADINA	55,000	12%	6,600
14/10	23001	43008	ALCHOLERA GRANADINA	55,000	5%	2,750
			TOTAL I.V.A. REPERCURTIDO:			83,810

con lo cual sí que no perderemos ni un solo bit.

La actualización de los ficheros del cuaderno de apuntes y su puesta a cero se ejecuta con la F6, antes de proceder a la cual, el ordenador nos avisa de que se van a perder los ficheros y nos pide confirmación. Realizada ésta podemos seguir introduciendo apuntes hasta agotar de nuevo la capacidad.

Instrucciones

La extremada sencillez del software del IVA, hace casi innecesario el uso de instrucciones, de todas formas y para la toma de contacto del usuario inexperto el programa está acompañado por 10 páginas de instrucciones en las que se describe el manejo del mismo de principio a fin.

En ellas se tocan todas las opciones y operaciones para conseguir el óptimo rendimiento del software con explicaciones concretas y cortas. Cualquier usuario puede hacerse con el dominio del programa en dos horas de instrucción.

Recapitulación

ZELIG nos obsequia con un programa de manejo sencillo y de una eficacia y prestaciones grandes para su uso.

Con un amplio dossier de informes impresos y consultas por pantalla, nada escapará a nuestros ojos y podremos tener amplia información de nuestros movimientos de clientes y facturas.

La sencillez de manejo de las opciones y su natural reproducción de los procesos reales hace que su aprendizaje y posterior utilización no representen ningún impedimento para alguien que no haya tocado en su vida esas máquinas diabólicas llamadas ordenadores.

Como en todo este tipo de programas la rapidez y velocidad en el proceso de datos viene dada por el número de cuentas y apuntes con que estemos operando, resultando lenta la ejecución cuando se trabaja con cifras elevadas.

Es importante reseñar que al final de la utilización el programa realiza un cierre de ficheros, por lo que no debemos extraer el disco hasta haber concluido todo el trabajo y sin dejar al programa realizar las operaciones finales del disco, en este punto un corte fortuito de la energía puede estropear el trabajo de varias horas, las copias de seguridad nunca están de más.

Un programa sencillo y eficiente con el que Hacienda y las dichosas declaraciones trimestrales no nos darán más quebraderos de cabeza, a los potentes ordenadores del fisco, el contribuyente puede oponer la informática doméstica y vencer en la contienda de los números.

PAGINA 1

NOMBRE DE LA EMPRESA: INVESTIGACION SA — LIBRO REGISTRO DE FACTURAS EMITIDAS

PERIODO: OCTUBRE/OCTUBRE

FECHA	FACT.	COD.	CLIENTE	B. IMPONIBLE	TIPO	CUOTA
12/10	04560	43001	INDSUTRIAS COMERCIAL	40,000	12%	4,800
12/10	04560	43001	INDSUTRIAS COMERCIAL	40,000	5%	2,000
15/10	02300	43002	FOTOGRAFIA JULIAN GOMEZ	50,000	12%	6,000
15/10	02300	43002	FOTOGRAFIA JULIAN GOMEZ	50,000	5%	2,500
20/10	43001	43003	ESTUDIO ALFA PUBLICIDAD	100,000	12%	12,000
20/10	43001	43003	ESTUDIO ALFA PUBLICIDAD	100,000	5%	5,000
17/10	01222	43004	AGENCIA INFORMACION & CO.	35,000	12%	4,200
17/10	01222	43004	AGENCIA INFORMACION & CO.	35,000	5%	1,750
08/10	43005	43005	IDIOMAS EL INGLES	70,000	12%	8,400
08/10	43005	43005	IDIOMAS EL INGLES	70,000	5%	3,500
19/10	04220	43006	ACADEMIA MATEMATICA Y CIA	23,000	12%	2,760
19/10	04220	43006	ACADEMIA MATEMATICA Y CIA	23,000	5%	1,150
19/10	55001	43007	CONDUCCIONES EL CANAL	120,000	12%	14,400
19/10	55001	43007	CONDUCCIONES EL CANAL	120,000	5%	6,000
14/10	23001	43008	ALCHOLERA GRANADINA	55,000	12%	6,600
14/10	23001	43008	ALCHOLERA GRANADINA	55,000	5%	2,750
			TOTAL I.V.A. REPERCURTIDO:			83,810

PAGINA 1

NOMBRE DE LA EMPRESA: INVESTIGACION SA

PERÍODO: OCTUBRE/OCTUBRE — COD: 40004 PROVEEDOR: DECORACION AMBIENTE SA.

FECHA	FACT.	COD.	PROVEEDOR	B. IMPONIBLE	TIPO	CUOTA
12/10	03221	40004	DECORACION AMBIENTE SA.	75,000	12%	9,000
12/10	03221	40004	DECORACION AMBIENTE SA.	75,000	5%	3,750
			TOTAL I.V.A. SOPORTADO:			12,750

DOMINE EL CODIGO MAQUINA EN SU AMSTRAD

Toda la gente que se interesa por el código máquina, ha visto, alguna vez, sus deseos frustrados por la «terrible complejidad», que aparentemente arrastra el aprendizaje de tan árido lenguaje, esto no es en absoluto cierto, y se debe a la poca visión que dan del código máquina algunos libros, que enseguida profundizan en el mismo, dejando al lector totalmente aturdido por la avalancha de rutinas, en su mayoría complejas, que intentan hacer comprender al, todavía, novato, poniendo a favor lo, poco clara, simplicidad de las mismas.



El libro de Gifford y Vincent intenta corregir estas deficiencias, explicando sin grandes pretensiones, la manera de comenzar a programar en código máquina, su texto es fácil de leer, dejando claros casi todos los conceptos que vamos viendo a medida que avanzamos.

El prólogo de los autores deja claro que es una guía y como tal no se extiende demasiado con cada nemotécnico, pero deja una visión clara de lo que hacen. Algunos serán difíciles de comprender en un principio, no importa, como todo costará un par de lecturas más de todo el libro y esos conceptos quedarán suficientemente claros.

El comienzo del libro es una breve explicación de qué es, para qué sirve y cómo programar código máquina, contando qué son los sistemas numéricos, por qué utilizarlos, qué es el lenguaje ensamblador. Incluso nos proporciona una pequeña muestra de los ensambladores disponibles para el **Amstrad,** casi todos publicados en Inglaterra, una lástima. Seguimos avanzando dándole un breve repaso al hardware, las llamadas a subrutinas en máquina, y a todos los demás conceptos básicos del código máquina.

Conviene resaltar el capítulo 8, ya que está dedicado a diccionario de términos de código máquina, dando una completa explicación de todos los registros, así como de los nemotécnicos de la Z80.

La explicación de cómo manejar los bits en código máquina, está hecha con gran claridad, dando una detallada explicación de cómo utilizar las instrucciones dedicadas al manejo de los mismos que posee la Z80.

Tenemos un capítulo dedicado a la pantalla del **Amstrad,** en el cual se hecha de menos algo más de extensión, ya que se dedica a comentar cómo funciona muy escuetamente, dando al mismo tiempo una serie de llamadas al firmware que apenas vienen a cuento.

Pero nadie es perfecto y está claro que Gifford y Vincent no iban a ser una excepción.

Más de la mitad del libro está dedicado a los listados, en basic y ensamblador de diversas rutinas en código máquina en las cuales abundan las de pantalla típicas, macrocaracteres, giro de frases a derecha e izquierda, compresores de pantalla.

Hay que destacar la rutina para mover un bloque, ya sea gráficos o texto, por la pantalla punto a punto, consiguiendo una suavidad de movimiento excepcional.

También se suministra un monitor de código máquina, que nos permitirá modificar la memoria a nuestro gusto.

Estas rutinas se entregan como utilidades, sin ningún ánimo de que sean entendidas a la primera por el lector.

Es un libro que merece la pena leer, ya que nos aclara bastantes conceptos de una manera amena.

Los autores han conseguido que éste sea un libro fácil de entender que se lee rápidamente, algo que no es nada fácil.

TITULO: DOMINE EL CODIGO MAQUINA EN SU AMSTRAD
EDITORIAL: RAMA
PRECIO: 2.200

APRENDE LOGO CON AMSTRAD

Logo es un lenguaje del que nos gustaría saber algo más, ya que la información que nos suministra con nuestro ordenador es bastante deficiente, por eso no es de extrañar que cuando nos ponemos a ver libros de informática y encontramos uno relacionado con este lenguaje, se nos vayan los ojos detrás suyo, esperando encontrar algo nuevo.

Nosotros hemos encontrado algo muy nuevo, Logo en Castellano. Un lenguaje como éste, creado y dedicado a la educación, no es muy adecuado que tenga las instrucciones primitivas en inglés, ya que los que no se lleven bien con dicha lengua, tendrán muchas más dificultades a la hora de intentar aprender Logo.

Pensando en los pobres niños a los que estamos intentando enseñar a programar con este lenguaje, cómo se tienen que liar al intentar codificar un programa utilizando un idioma que no es el suyo, esto no es muy didáctico que digamos.

Bien, a Spen, S.A., se le ocurrió la idea de hacer que el Logo del **Amstrad,** porqué no, estuviera en castellano, esto se consigue mediante una serie de primitivas definidas en un fichero. No es que sea algo muy complicado, pero a ellos se les ocurrió primero.

El libro está enfocado a los niños, por tanto no se mete a explorar las múltiples posibilidades de este lenguaje, cosa que se le podía ocurrir a alguien, tal vez **Amstrad** España se digne a publicar la Guía de Logo que debiera acompañar al intérprete suministrado con el ordenador. El libro no descubre nada que no sepa el que se haya leído el manual. Pero al niño se le explican claramente casi todas las primitivas, en castellano, de las que disponemos; es una magnífica guía infantil de este lenguaje.

TITULO: APRENDE LOGO CON AMSTRAD
EDITORIAL: RAMA
PRECIO: 2.650

Las primitivas de las cuales disponemos en castellano son las siguientes:

SI	IF
ATRAS	BK
IZQUIERDA	LT
DERECHA	RT
ADELANTE	FD
BORRAR	PE
EMPEZAR	CS
LISTAR	POTS
VER	ST
CARGAR	LOAD
SALVAR	SAVE
HAZ	MAKE
ESCRIBIR	PR
ESPACIO	NODES
UNE	WORD
ADIOS	BYE
PUNTO	DOT
SONIDO	SOUND
ALEATORIO	RANDOM
EDITAR	ED
ENTRADA	RQ
DIRECTORIO	DIR
REPETIR	REPEAT
CUENTA	COUNT
PLUMA-VISIBLE	PD
VER-TORTUGA	ST
ESCONDER-TORTUGA	HT
BORRAR-VARIABLE	ERN
COLOR-TORTUGA	SETPC
PANTALLA-DE-GRAFICOS	FS
VUELVE-TORTUGA	WRAP
SIN-PRIMERA	BF
SIN-ULTIMA	BL
PANTALLA-DE-TEXTO	TS
LISTAR-PROCEDIMIENTO	PO
DEJAR-ESPACIO	RECYCLE

Aquéllas que no encontremos no nos será muy difícil implementarlas mediante la primitiva adecuada.

Podemos utilizar el Logo nativo de **Amstrad** mezclado con el Logo Castellano, ya que no influye, al ser procedimientos, el que hallamos cargado el castellano.

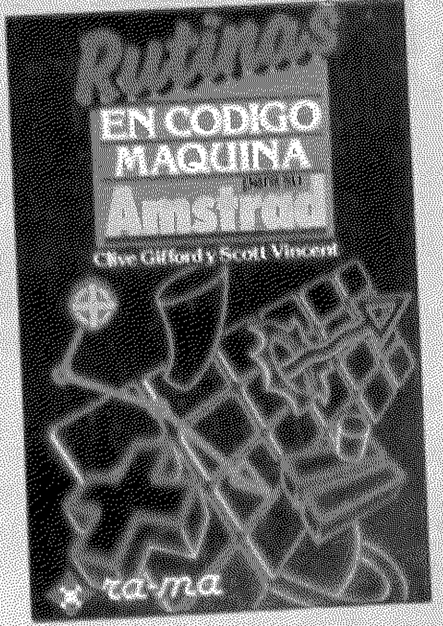
El libro viene repleto de ejercicios, solucionados y sin solucionar, que ayudan a una mejor comprensión de los conceptos en él explicados. Todos los ejemplos vienen acompañados por sus dibujos correspondientes, cosa que aclara mucho qué es lo que hace esa tarea, una imagen...

En definitiva, el niño puede aprender a manejar Logo con este libro, que fundamentalmente es de los que se trata, el mayor puede aclarar algunos conceptos oscuros, pero no creáis que todos.

RUTINAS EN CODIGO MAQUINA PARA SU AMSTRAD

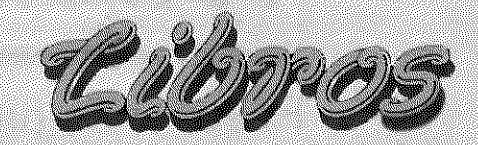
El código máquina es un tema sobre el que existen libros para todos los gustos y niveles. Desde los que comienzan mostrando al lector la definición de lo que es un bit, hasta los que en su segunda página poseen ya kilométricos listados de ensamblador y complicadas definiciones de firmware.

Una de las cosas que uno ha de tener muy presente, dada la amplia gama de libros que como ya decimos existe sobre el tema, al comprar un libro de código máquina es el nivel de conocimientos que posee y la utilización que va a hacer de él.



No es lo mismo comprar un libro de listados y subrutinas que nos ayuden a hacer más rápidos nuestros programas en basic que comprar un libro de código máquina con el deseo de aprender las reglas por las cuales se rige su programación.

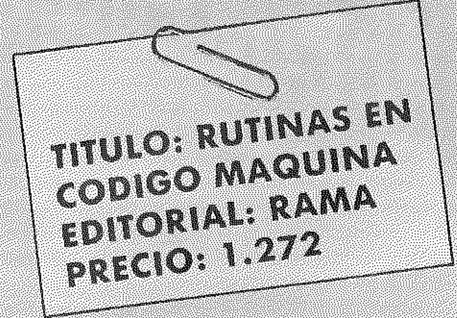
En el caso de este libro la elección está muy clara; es recomendable a todo aquél que quiera aumentar la potencia de sus programas basic, dándoles rapidez y efectos gráficos que en basic es imposible conseguir.



Desde la primera página el libro se compone de interesantes subrutinas para el manejo de pantallas, sonido, etc. Algo de agradecer: las subrutinas vienen dadas primero en basic, introduciéndolas en el ordenador mediante una serie de pokes.

Luego aparece el listado en ensamblador, con lo cual el lector que sienta ese natural respeto que todos sentimos al principio por estos listados puede eludirlos tranquilamente e introducir la subrutinas vienen dadas primero en basic, introduciéndolas en el ordenador mediante una serie de pokes. Luego aparece el listado en ensamblador, con lo cual el lector que sienta ese natural respeto que todos sentimos al principio por estos listados puede eludirlos tranquilamente e introducir la subrutina «pokeando» como ya dijimos antes.

Por el contrario, el lector que sepa qué labor desempeña cada uno de los nemónicos y conozca bien los registros del Z-80 puede examinar los listados de ensamblador e intentar sacar algo en claro para las subrutinas que luego él mismo cree.



Como ya hemos comentado, casi todas las subrutinas van dirigidas a hacer verdaderas maravillas con lo que tenemos en pantalla.

Una pequeña muestra de éstas: inversor de pantalla, copiador de texto, desplazamiento de texto, desplazamiento de bloques, explosión gráfica, letras itálicas, inclinadas, y, como se suele decir un largo etc.

En resumen, este libro pertenece a los que no tienen por intención enseñarnos el código en sí, sino darnos unas cuantas subrutinas para agilizar nuestros programas, tarea que desempeña perfectamente.

Por último algo que debería ser trivial pero que por desgracia no lo es en algunas ocasiones, las subrutinas del libro funcionan.

SUSCRIBETE POR UN AÑO Y PAGA SOLO 8 MESES.

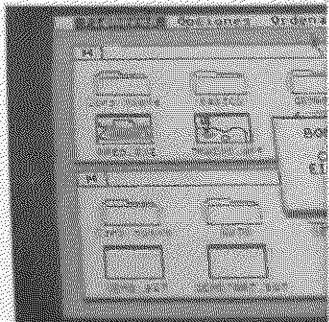
Ahórrate 3.500 ptas.: un 35 %.



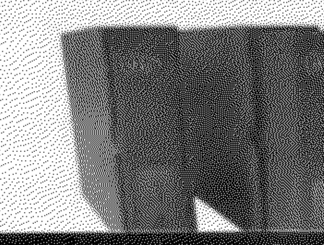
REVISTA INDEPENDIENTE PARA USUARIOS DE ORDENADORES AMSTRAD

TODOS LOS SECRETOS DEL PC 1512

Lo que hay que saber del nuevo ordenador de Amstrad: sus características, ventajas e inconvenientes. Además, un exhaustivo análisis de todas las aplicaciones GEM: programas, sistemas operativos y mucho más.







HOBBY PRESS.

AMSTRAD

También puedes suscribirte por teléfono (91) 734 6500

CUPON DE SUSCRIPCION

SI, deseo suscribirme a la revista **Amstrad Semanal** por un año (50 números), al precio de 6.175 ptas., lo que me supone **un ahorro de casi 3.500 ptas.** (oferta válida sólo para España, hasta el 28 de febrero de 1987).

Nombre ____________________

Apellidos ____________________

Fecha de nacimiento ____________________

Domicilio ____________________

Localidad ____________ Provincia ____________

Teléfono ____________ C.P. ____________

Para agilizar tu envio, es importante que indiques el Código Postal.

Forma de pago:

☐ Talón a nombre de Hobby Press, S. A.

☐ Giro postal a nombre de Hobby Press, S. A., n.º ____________

☐ Contra reembolso (supone 100 ptas. más de gastos de envio y es válido sólo para España).

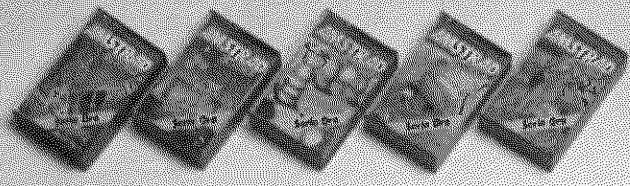
☐ Mediante tarjeta de crédito

n.º ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐

☐ Visa ☐ Master Charge ☐ American Express

Fecha de caducidad de la tarjeta ____________

Fecha y firma



CUPON DE PEDIDO DE NUMEROS ATRASADOS Y CINTAS SERIE ORO.

SI, deseo recibir en mi domicilio los siguientes ejemplares de **Amstrad Semanal,** cuyos números indico a continuación, al precio de 190 ptas. cada uno (independientemente del precio que aparezca en portada y de su fecha de publicación). Separo con un guión cada número ____________________

Está agotado el número 4.
Si compro más de siete números recibiré, totalmente gratis, una cinta de Your Computer.

SI, deseo recibir en mi domicilio las **cintas Serie Oro** que indico a continuación, al precio de 756 ptas. cada una. Cada cinta lleva grabados los programas publicados en cuatro números de la revista (1 al 4, 5 al 8, 9 al 12, etc.).

Números ______ al ______ Números ______ al ______

Números ______ al ______ Números ______ al ______

Números ______ al ______ Números ______ al ______

Nombre ____________________

Apellidos ____________________

Fecha de nacimiento ____________________

Domicilio ____________________

Localidad ____________ Provincia ____________

Teléfono ____________ C.P. ____________

Para agilizar tu envio, es importante que indiques el Código Postal.

Forma de pago:

☐ Mediante talón bancario a nombre de Hobby Press, S. A.

☐ Mediante giro postal a nombre de Hobby Press, S. A., n.º ____________

☐ Contra reembolso (supone 100 ptas. más de gastos de envio y es válido sólo para España).

No se admiten solicitudes de cintas con pago contra reembolso.

Fecha y firma

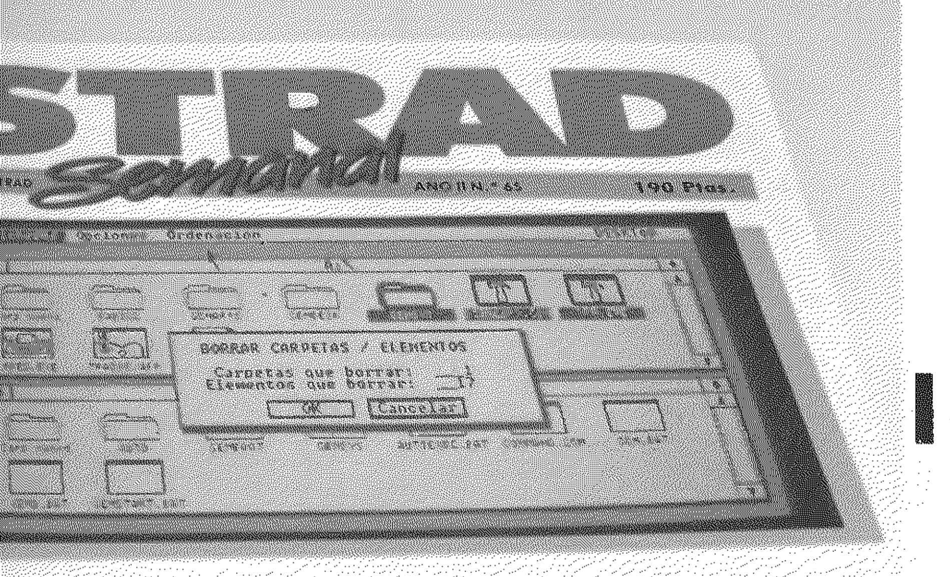


Esta es la oferta del año, todo un año de AMSTRAD Semanal por sólo 6.175 ptas. 50 números que te salen a un precio increíble: 123 ptas. cada uno.

un 35%.

Aprovéchate. Una oferta así sólo se presenta una vez al año.

(oferta válida sólo para España, hasta el 28 de febrero de 1987).



ESS. Para gente inquieta.

RESPUESTA COMERCIAL
Autorización n.º 7427
B.O.C. y T. n.º 81
de 29 de agosto de 1986

HOBBY PRESS, S.A.
Apartado n.º 8 F.D.
28100 Alcobendas
(Madrid)

No necesita sello. A franquear en destino.

RESPUESTA COMERCIAL
Autorización n.º 7427
B.O.C. y T. n.º 81
de 29 de agosto de 1986

HOBBY PRESS, S.A.
Apartado n.º 8 F.D.
28100 Alcobendas
(Madrid)

No necesita sello. A franquear en destino.